Benchmarking openGauss Against PostgreSQL
A Comparative Analysis

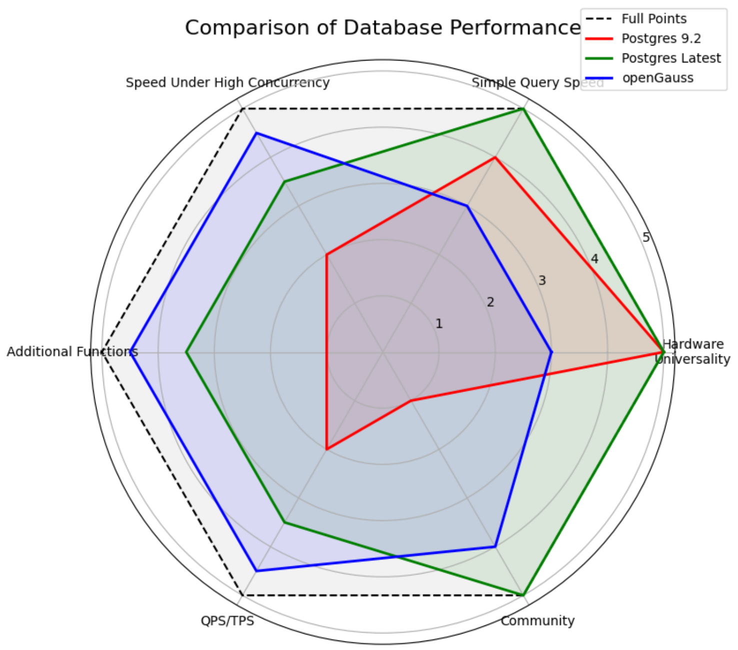
Jingqi SUN

Southern University of Science and Technology

Department of Computer Science

URL: https://github.com/JingqiSunn/Benchmarking-openGauss-Against-PostgreSQL-A-Comparative-Analysis.git

December 2024



My project is a little bit long, so I write a abstract section and put it in the front.
This abstract section can be seen as the major part of my report, it contains the graphic result of all my experiments, my experiments settings and the conclusion that I can draw from it.
If you have any doubt about any part of this section, you can look over the corresponding section for each experiment, there will contain my pseudo code, my design, my task and table of data.
If you want to see the source code, they are in the code section.
Best Wishes

Contents

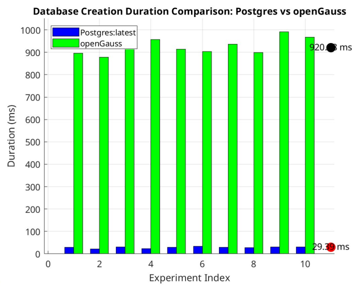
1 Abstract

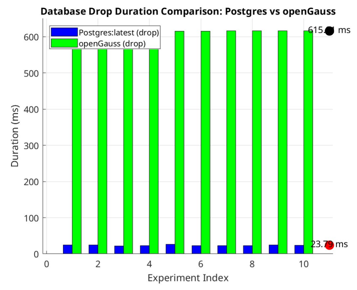
1.1 Evaluation Criteria

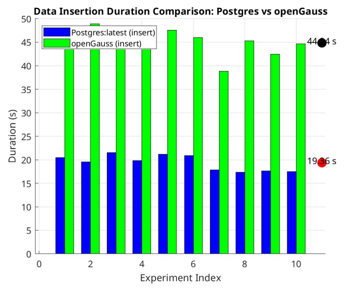
- speed to do a simple query like INSERT, SELECT
- speed to do query under situation of high concurrent
- QPS/TPS under situation of high concurrent
- error rate under situation of high concurrent

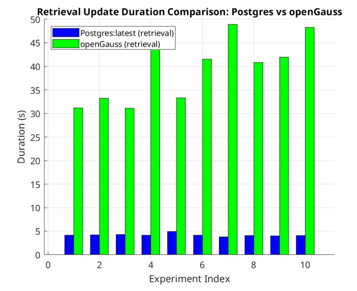
1.2 Final Conclusion

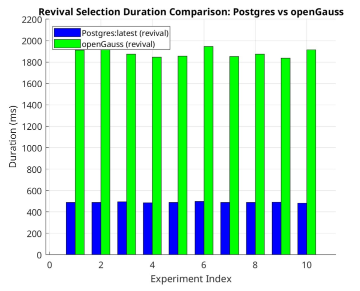
- OpenGauss is query-wise slow on my computer with default configeration compared to postgresSQL:latest

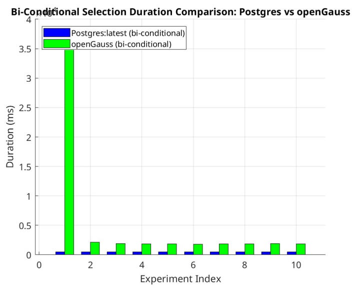


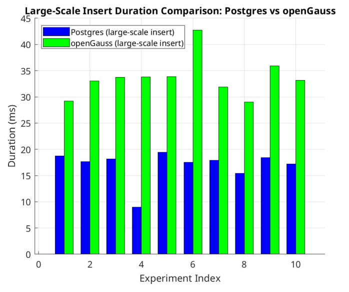




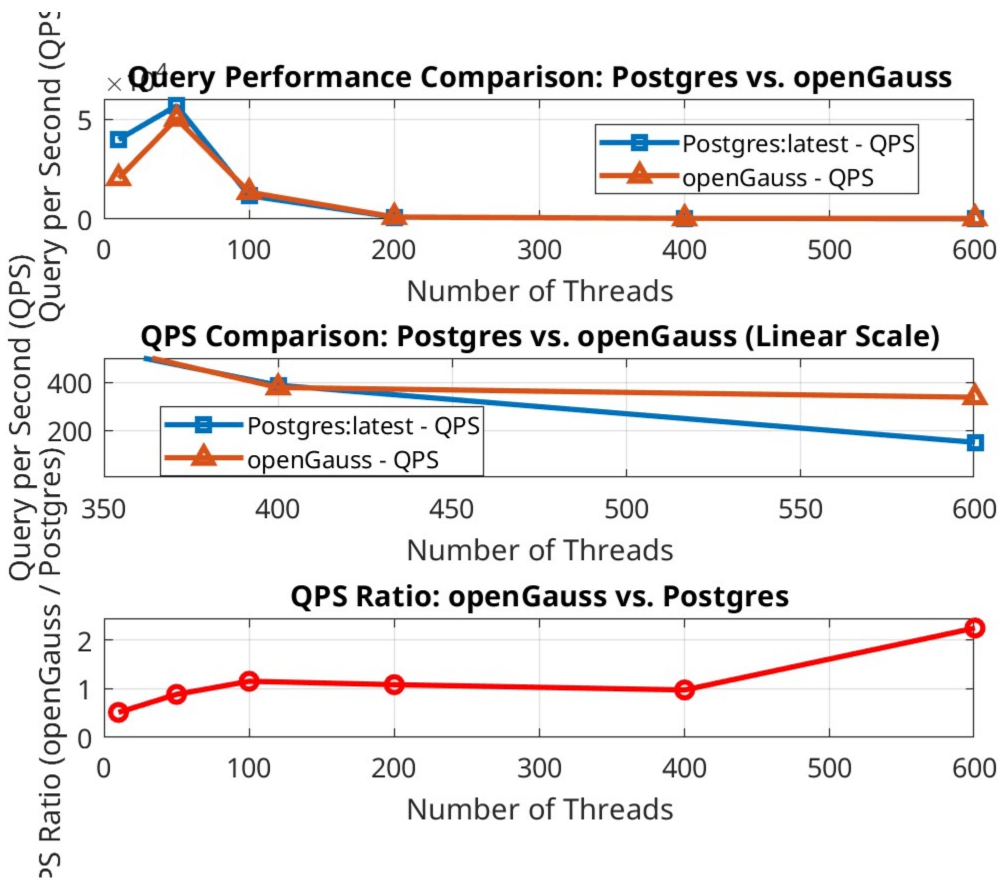




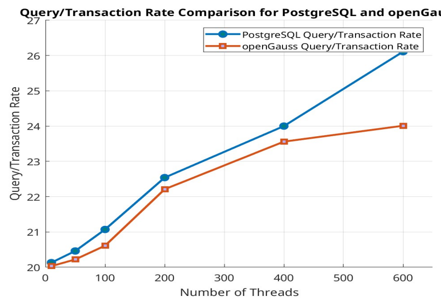




- OpenGauss has a potential to have a huge advantage on QPS under the situation of pretty high concurrency



- OpenGuass can relatively keep a low value of QPS/TPS under situation of high concurrency compared to postgres:latest



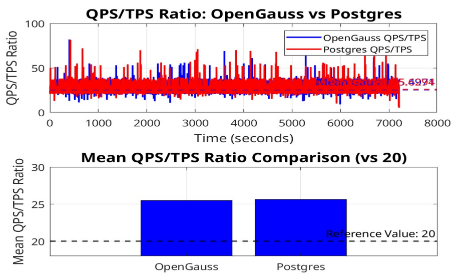
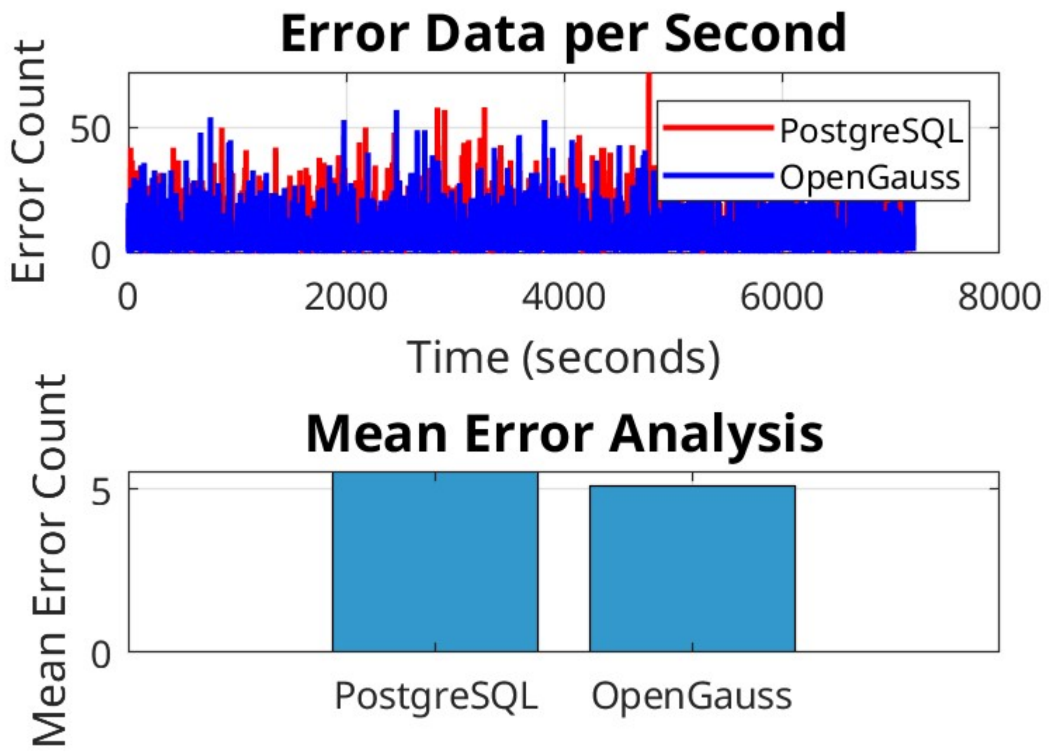


Figure 1: Your caption here

- OpenGauss has a potential to keep error rate relatively low compared to postgres:latest under situtaion of high concurrency.

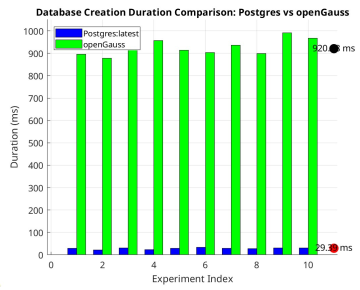


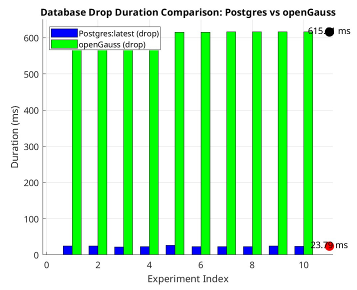
1.3 Recall phenomenon I encountered all the ways In order

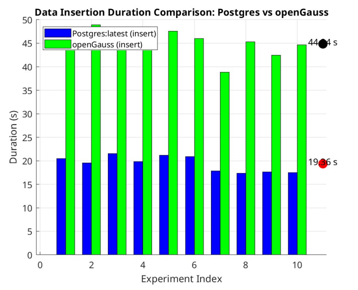
1.3.1 OpenGauss is Query-wise Slow on my Computer x86_64 13th Gen Intel(R) Core(TM) i9-13900HX

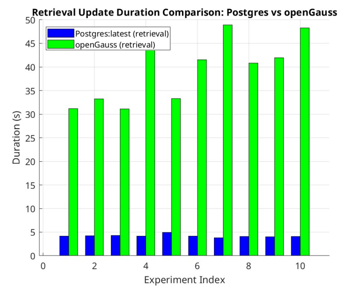
When I get this project, the first thing comes to my mind about the ability of a database is how fast can it do some basic operations like CREATE DATABASE, INSERT, SELECT... I mean, what is the operation time.

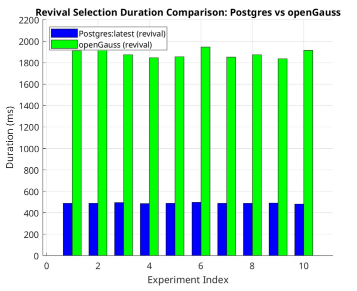
So I did some test on basic operations based on the NBA database. The result indeed shocked me a lot.(The detail information about these experiments are in the chapter "Basic Experiments")

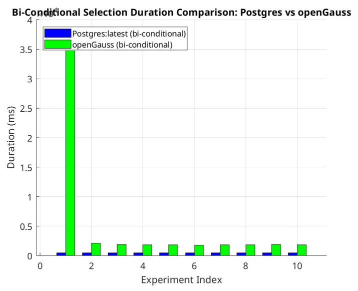


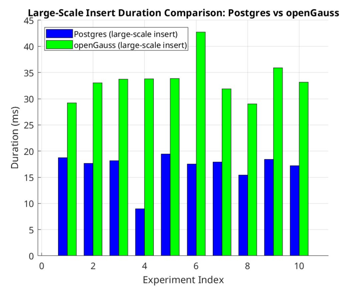










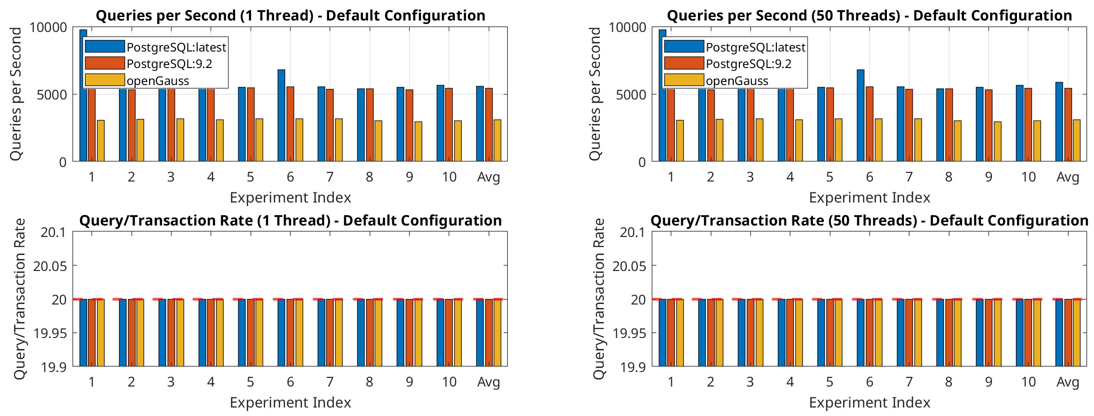


As you can notice as the graph shows, **it takes extremely long time for openGauss to do some basic queries compared to postgres:latest** with default configeration.

1.3.2 OpenGauss has potentail to keep QPS relatively high under high concurrent & postgres:latest has similar performance to postgres:9.2 on most tasks & OpenGauss is even Slower than postgres:9.2 on basic low recurrence tasks

After the shocking result from the previous tasks on basic query execution speed, I got the background information that **OpenGauss was developed based on postgres:9.2**, so my choise is that in the following test, we will also test postgres:9.2

I did five tests on these three databases, the first two tests is on default configuration. **I use sysbench to test performance of these three databases**, my original purpose is to use this testbench to vertify that whether OpenGauss is as slow as what I find in the previous task. So we got the result.(The detail of these test are in the chapter "Basic Experiment on sysbench")



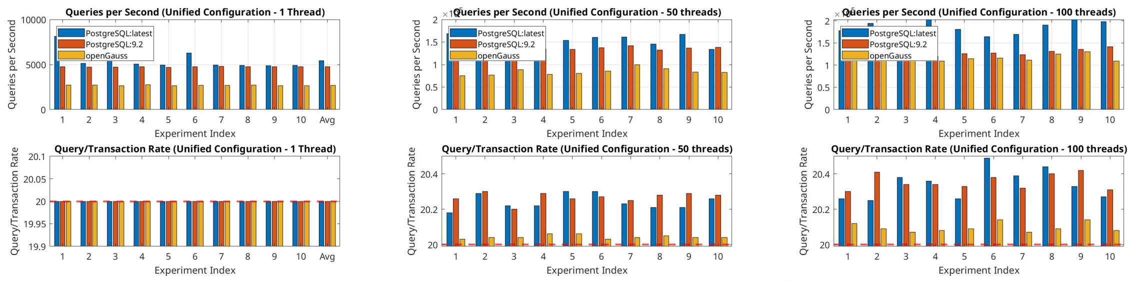
I recorded three data totally

- QPS(query per second)
- TPS(transection per second)
- QPS/TPS

And we can find from this result that **postgres:9.2 has similar performance to postgres:latest,OpenGauss is indeed SLOW under low concurrency**.

What's more, because these two tests actually did not put too much pressure on the database, so we can see that QPS/TPS is optimally 20.

After these findings, I have another guess, is it because the difference in default configuration that makes OpenGauss loss so much? With this thought, I entered the dockers and unified all the important configuration about resource limitations. Then we did three tests.(The detail of these test are in the chapter "Basic Experiment on sysbench")



There are two parts of the result, one is **although this time there is noticable difference between postgres:latest and postgres:9.2, when it is dealing with small concurrent like 1 and 50, openGauss is still really SLOW**

But there is another new part of the result which gives me hope to find the advantage of OpenGauss. We can notice that

- **Although OpenGauss is the slowest in all three tests, but its relative performance actually improved with more concurrency**, which means that OpenGauss might be really good at handle large concurrency.
- **In the former two tests, we might notice that QPS/TPS is always a constant 20 reveal the optimal solution to the sysbench, but now, all the numbers are starts to get over 20,**

luckily postgres explodes much faster than OpenGauss, which indicate that the optimizer in OpenGauss can undergo more pressure with high concurrency and make the better optimaize decision.

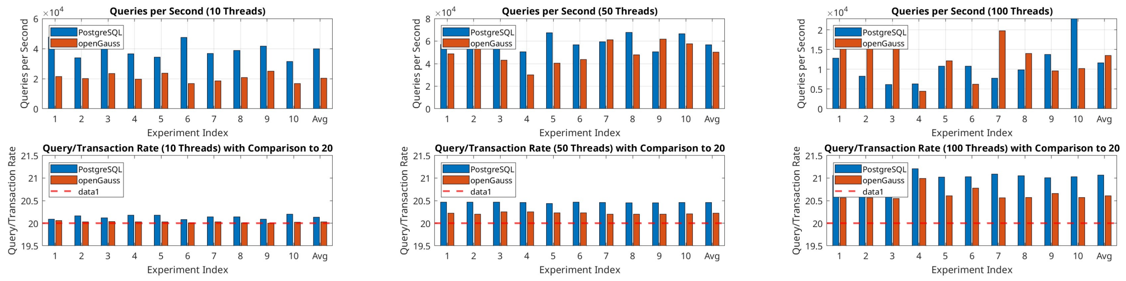
With these two thoughts, I designed the next few experiments.

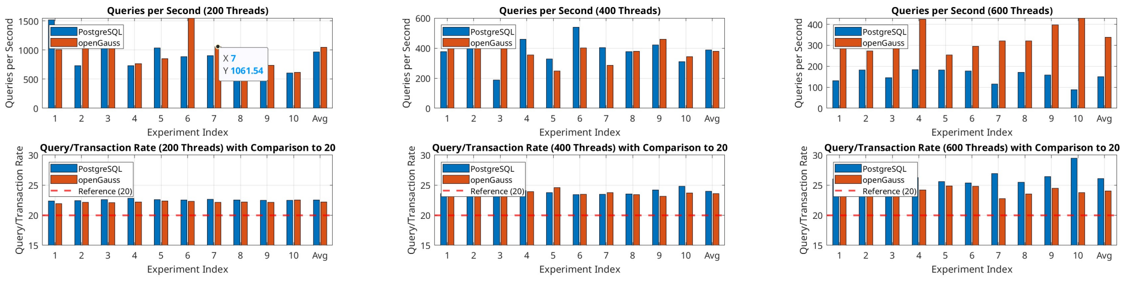
1.3.3 OpenGauss has ability to maintain relatively high QPS Under large concurrency relative to postgres:latest

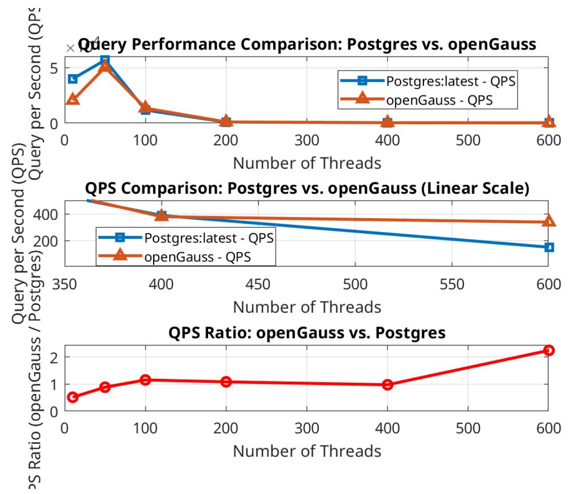
In the previous experiments, we see that OpenGauss might have good performance with large concurrency based on our observation on **QPS changes along with the change of threads number and QPS/TPS**.

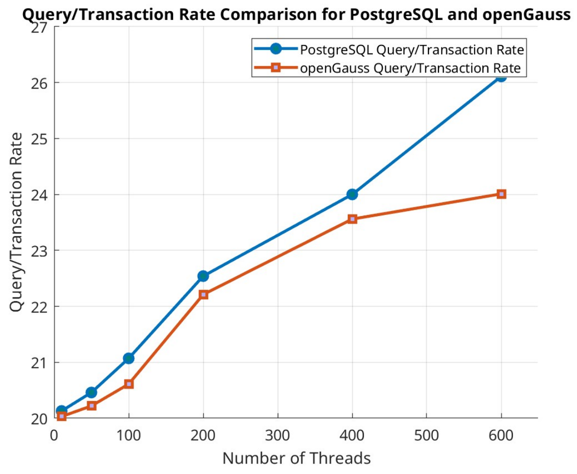
And here I encounter a problem that the version of debian system for postgres:9.2 is too old to apt-get install vim or any other text editor, I have no choice but to only take tests on postgres:latest and OpenGauss. But we have already got the information that when threads number is low, postgres:9.2 has similart performance to postgres:latest, but when threads number gets high, there will be potential to have huge difference.

I reunified the configuration parameters of Postgres:latest and OpenGauss, and I use sysbench to do some tests mainly about the concurrency.







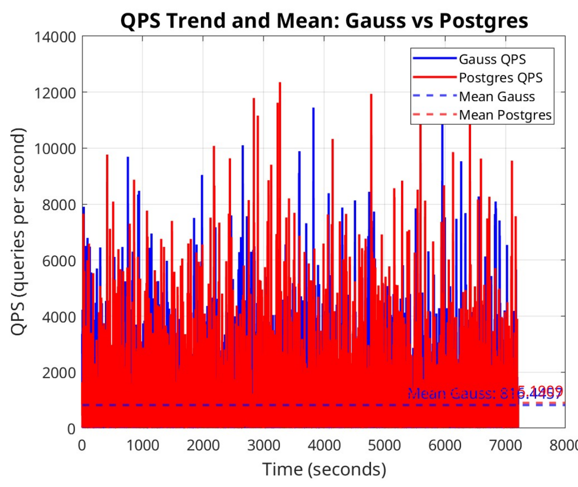


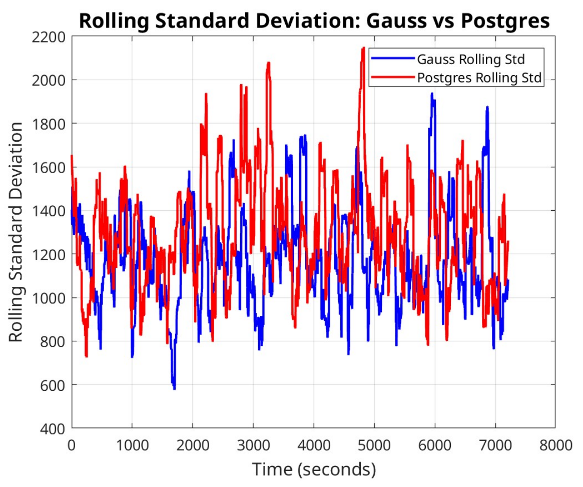
From the graph we can notice that our two guess about **QPS and QPS/TPS** on the last test are verified. In the graph, it is clear that **when concurrency is low, Postgres:latest can do more query than OpenGauss, but when threads number goes high, we can find that at thread = 600, we have OpenGauss can do twice number of query per second compared to postgres:latest**, and also we can find that **QPS/TPS expand faster for postgres:latest. All of these data is powerful evidence yelling that OpenGauss does better under high concurrency.**

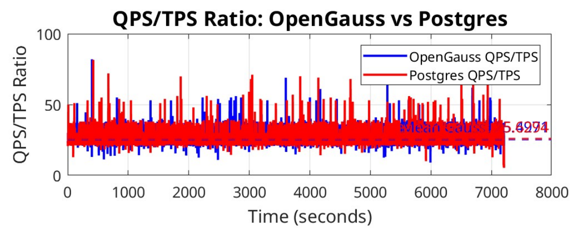
1.3.4 OpenGauss has Lower Error rate than postgres:latest, which also means some kind of stability

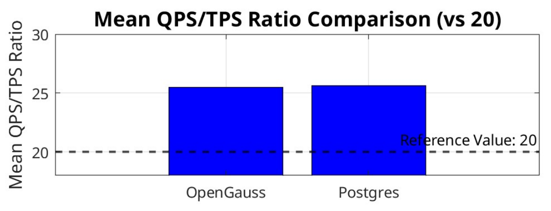
After I give the conclusion OpenGauss can have better performance than OpenGauss, I think it is time to test the stability of the database under high concurrency.

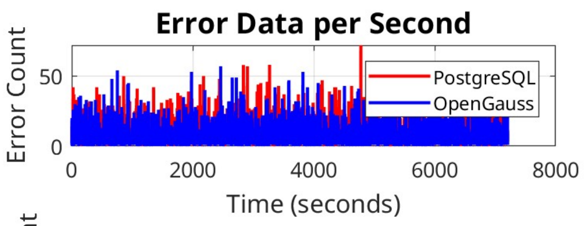
So still using the sysbench, I run 2 hour 500 threads 3 table test on each databse(I mean to use 600 threads first, but for some config problem that I can not solve, openGauss will always crash, and I don't think it has anything to do with the database, it is more about the configuration and docker limit, but I can not fix it that time.)

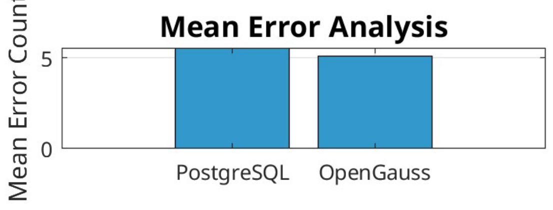












And it is just like the result of the last problem, we can not see significant different in QPS because it is in some kind of edge. But we can notice that **The deviation of QPS of OpenGauss is smaller than Postgres and also it has smaller QPS/TPS**, these two thing that although it is not in 600 threads, but OpenGauss is going to take over Postgres on speed and stability. Also we can notice that **the error of OpenGauss is smaller**, which is also a criteria to say that openGauss has better performance on high concurrency.

Unfortunately, limited on the equipment and time, I can not have the chance to do more extreme test on postgres and OpenGauss, 600 threads is obviously not an extremely high concurrency. But from our tests, we can indicate that openGauss can have excellent High concurrent performance compared to OpenGauss, because there has already been strong evidence and trend for that.

1.4 Analyze some conclusion

From the previous sections, we can draw lots of conclusions on the performance of OpenGauss and postgres:latest also the postgres:9.2.

- OpenGauss is slow on simple query

  I don't know the exact reason, but I guess that is mostly about the hardware(maybe $\times 86$ system is not the best system for it) and config.

- OpenGauss is better under High concurrence

  We all know that we can not speed up the execution of a single query, no matter how large the concurrency is, the minimum time to finish a simple query for a database will never been changed, and OpenGauss is as slow as before to execute a query. However, it seems that the transactions system in the OpenGauss can be more efficient and make less mistake, being less messy under big concurrency, which leads to its better performance under high concurrency.

  I guess the reason can be

  - AI Query Time Forecasting
  - parallel query
  - Dynamic Build and Execution
  - CBO Optimizer

2 Data Implementation

2.1 Download Data

I import my dataset from https://www.kaggle.com/, the original dataset is in the form of .csv, with size about 2.3 GB, for the reason of convenience, I set all the keys to be string at first.

2.2 Importing .csv file to postgres

I used the code in 'import.sql' which creates several table in the main schema of my database DBMSPerformanceEvaluation.

2.3 Explore Basic Information of my Data

I used the code in 'count_rows_columns.sql' which count the number of size, rows and keys of my database.
And here is the result

table	size	rows	columns
common_player_info	1200kb	4171	33
draft_combine_stats	264kb	1202	47
draft_history	1112kb	7990	14
game	22mb	65698	55
game_info	4136kb	58053	4
game_summary	7976kb	58110	14
inactive_players	11mb	110191	9
line_score	12mb	58053	43
officials	4648kb	70971	5
other_stats	4400kb	28271	26
play_by_play	2539mb	1359289	34
player	360kb	4831	5
team	16kb	30	7
team_details	16kb	25	14
team_history	16kb	52	5
team_info_common	8192b	0	26

2.4 Test Bench

I choose sysbench to be my test bench in some of my chapters.

3 Timing Standard

In the chapter of Basic experiments, these DIY tests are based on the timing command in the database, and in the par of sysbench, I will use the information provided by the benchmark itself.

4 Basic Experiments

4.1 Create Database

4.1.1 Design

To start the whole test project, I think the first necessary thing is to examine the time to create a database or drop a database. So in this task, I decide to compare the speed for postgresSQL and openGauss to create a new database called "DBMS".

4.1.2 Implementation

Here is my pseudo code

```
DO_OPERATION_WITH_TIMING()
1: CREATE DATABASE DBMS;
```

4.1.3 Result

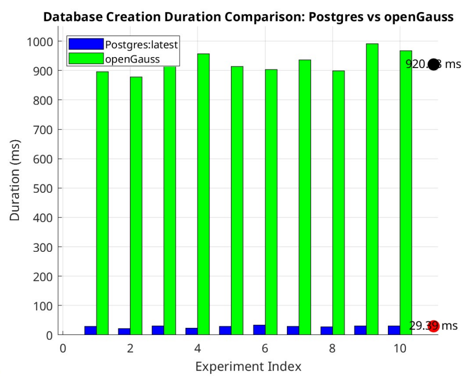
I run the code 'CREATE DATABASE DBMS' for 10 times, and after each time I will do the DROP operation.
here is the result for postgresSQL.

experiment_index	duration
1	29.55ms
2	22.28ms
3	30.71ms
4	22.76ms
5	29.60ms
6	32.90ms
7	29.24ms
8	27.33ms
9	30.33ms
10	29.99ms
average	29.39ms

here is the result for openGauss.

experiment_index	duration
1	895.92ms
2	878.06ms
3	951.41ms
4	957.37ms
5	913.65ms
6	902.83ms
7	935.75ms
8	898.62ms
9	991.68ms
10	967.39ms
average	920.58ms

We will have the graph to be



4.2 Drop Database

4.2.1 Design

In this test, we will examine the speed for both database system to drop a dababase.

4.2.2 Implementation

Here is my pseudo code

```
DO_OPERATION_WITH_TIMING()
1: DROP DATABASE DBMS;
```

4.2.3 Result

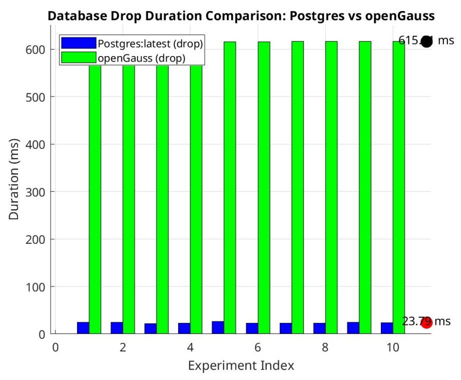
I run the code 'DROP DATABASE DBMS' for 10 times, and before each time I will do the CREATE operation.
here is the result for postgresSQL.

experiment_index	duration
1	24.23ms
2	24.53ms
3	21.57ms
4	22.46ms
5	26.58ms
6	23.03ms
7	22.41ms
8	22.52ms
9	24.02ms
10	23.12ms
average	23.79ms

here is the result for openGauss.

experiment_index	duration
1	615.34ms
2	615.01ms
3	616.92ms
4	615.74ms
5	615.38ms
6	615.53ms
7	615.96ms
8	616.41ms
9	616.46ms
10	616.41ms
average	615.91ms

We will have the graph to be



4.3 Data Insertion

4.3.1 Design

After testing the time to create and drop a database, I think it is time to insert the data into the postgresSQL and openGauss.
And before the test, I have already created a database in each of the system called 'DBMS'.

4.3.2 Implementation

Here is my pseudo code

```
DO_OPERATION_WITH_TIMING()
1: FOR several tables
2: IF already exits THEN DROP it
3: CREATE TABLE
4: IMPORT data FROM corresponding CSV file
```

4.3.3 Result

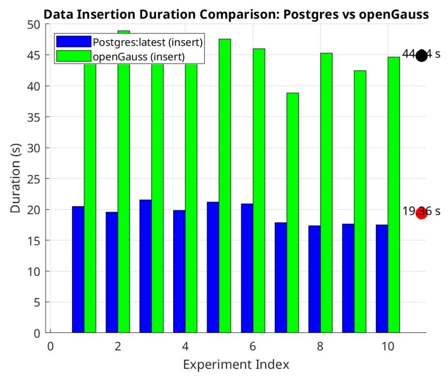
I will run the code in the 'import.sql' to do the data insertion.
Here is the result of postgresSQL.

experiment_index	duration
1	20.49s
2	19.51s
3	21.51s
4	19.81s
5	21.17s
6	20.86s
7	17.84s
8	17.36s
9	17.65s
10	17.47s
average	19.36s

here is the result for openGauss.

experiment_index	duration
1	45.70s
2	48.91s
3	45.08s
4	44.03s
5	47.52s
6	45.98s
7	38.84s
8	45.28s
9	42.44s
10	44.65s
average	44.84s

We will have the graph to be



4.4 Retrieval Selection

Task 4.1. *Select all the information of the player called "Kevin Love" in the table 'play_by_play'*

4.4.1 Design

According to the task, we have to select all the rows with 'player1_name' = 'Kevin Love' of the table 'play_by_play'. So I just use the SELECT operation in my sql code.

4.4.2 Implementation

Here is my pseudo code

```
DO_OPERATION_WITH_TIMING()
1: SELECT * FROM play_by_play where WHERE player1_name = 'Kevin Love'
```

4.4.3 Result

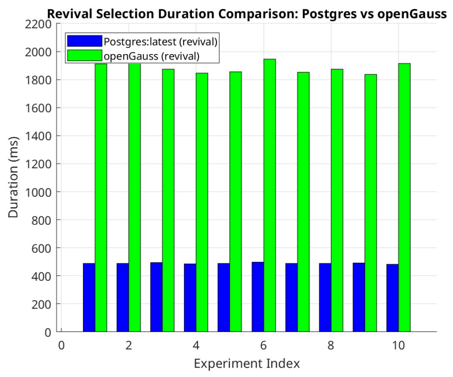
I will run the code in the 'retrieval_selection.sql' to do the data selection.
Here is the result of postgresSQL.

experiment_index	duration
1	488.155 ms
2	488.384 ms
3	494.223 ms
4	485.179 ms
5	489.532 ms
6	497.237 ms
7	486.928 ms
8	488.424 ms
9	492.507 ms
10	483.013 ms

here is the result for openGauss.

experiment_index	duration
1	1911.997 ms
2	2066.188 ms
3	1874.104 ms
4	1846.240 ms
5	1855.313 ms
6	1946.687 ms
7	1853.153 ms
8	1872.931 ms
9	1837.091 ms
10	1913.054 ms

We will have the graph to be



4.5 Retrieval Update

Task 4.2. *Change the name in the table 'play_by_play' column 'player1_name' from the form TO to the form of TTOO.*

4.5.1 Design

According to the task, I have to change "To" to "TTOO" in player1_name, and it is recommended to use UPDATE and REPLACE operation here.

4.5.2 Implementation

Here is my pseudo code

```
1:    Update play_by_play set player1_name to
      REPLACE(player1_name, 'To', 'TTOO')
2:    Where player1_name contains 'To'
```

4.5.3 Result

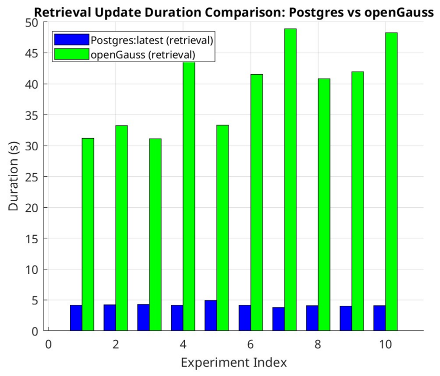
I run the code in the 'name_transformation.sql' for 10 times(everytime after I execute 'name_transformation.sql', I will run 'import.sql' to reset the table for operation). Here is the result for postgresSQL, the actual code I ran here is 'actual_retrieval_update.sql'

experiment_index	duration
1	4.13s
2	4.20s
3	4.28s
4	4.13s
5	4.96s
6	4.16s
7	3.82s
8	4.06s
9	4.01s
10	4.06s
average	4.18s

Here is the result of openGauss.

1	31.17s
2	33.21s
3	31.08s
4	46.13s
5	33.27s
6	41.51s
7	48.93s
8	40.83s
9	41.93s
10	48.24s
average	45.43s

We will have the graph to be



4.6 Bi-Conditional Selection

Task 4.3. *Select all the information of the player called "Kevin Love" in the table 'play_by_play' and the game_id is 0021600215.*

There is a brief explanation to this task setting. If we just do the selection by mono-condition, then we can see that

1. If we select by player1_name = "Kevin Love", there will be 29022 rows of output.
2. If we select by game_id = '0021600215', there will be 443 rows of output.

Considering the big difference in the amount of output data, it is suitable to do the following tests.

4.6.1 Design

According to the task, I have to select all the information of the player called ”Kevin Love” in the table ’play_by_play’ and the game_id is 0021600215.
Then I will simply use the SELECT command.

4.6.2 Implementation

Here is my pseudo code

```
SELECT * FROM play_by_play WHERE play1_name = 'Kevin Love'
AND game_id = '0021600215'
```

4.6.3 Result

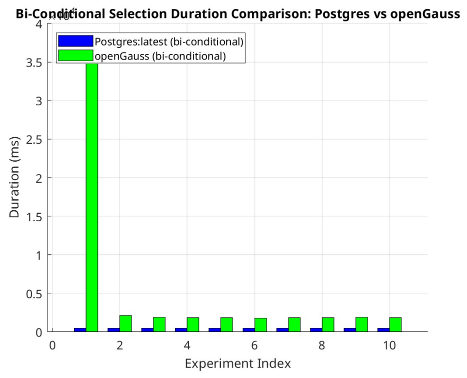
I run the code in the ’biconditional_selection.sql’ for 10 times.
Here is the result of postgresSQL.

experiment_index	duration
experiment_index	duration
1	491.428 ms
2	491.331 ms
3	486.887 ms
4	488.107 ms
5	493.474 ms
6	489.264 ms
7	490.287 ms
8	487.453 ms
9	487.436 ms
10	489.107 ms
average	489.077 ms

Here is the result of openGauss.

experiment_index	duration
1	37663.644 ms
2	2140.002 ms
3	1884.586 ms
4	1851.146 ms
5	1822.042 ms
6	1814.894 ms
7	1868.863 ms
8	1842.210 ms
9	1917.811 ms
10	1830.208 ms
average	5811.788 ms

We will have the graph to be



4.7 Large-scale Insert From other Table

Task 4.4. *I create a 20000 rows' csv file 'game_ data_ 20000_ rows.csv' that has the same entry format as the table game_ info, and I need to insert the new table into the database of the old table.*

For the convenience, I update my code 'import.sql' to create another table in my database called 'game_data_20000_rows', see the code 'insert_plus_plus.sql'.

4.7.1 Design

According to the task, I have to insert the new table into the database of the old table.
I can directly use SELECT to add all the rows from the table 'game_data_20000_rows' to the table 'game_info'.

4.7.2 Implementation

Here is my pseudo code

```
1: SELECT a row FROM TABE+LE game_data_20000_rows
2: INSERT INTO TABLE game_info
3: until there is no rows left.
```

4.7.3 Result

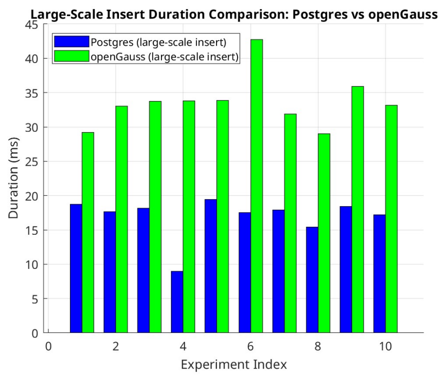
I run the code in the 'insert_table_by_select_table_big.sql ' for 10 times(Every time before I do the test, I have to run 'import_plus_plus.sql' to reboot the setting)
The code I actually run is 'actual_large_scale_insert.sql'
Here is the result of postgresSQL.

experiment_index	duration
experiment_index	duration
1	18.715 ms
2	17.661 ms
3	18.150 ms
4	8.986 ms
5	19.463 ms
6	17.504 ms
7	17.901 ms
8	15.409 ms
9	18.443 ms
10	17.205 ms
average	16.944 ms

Here is the result of openGauss.

experiment_index	duration
1	29.211 ms
2	33.055 ms
3	33.737 ms
4	33.792 ms
5	33.851 ms
6	42.757 ms
7	31.890 ms
8	29.022 ms
9	35.882 ms
10	33.152 ms
average	33.835 ms

We will have the graph to be



4.8 Conclusion

As you see in the above graphs and data, postgresSQL beats OpenGauss hard nearly in every tasks, it is meaning less to test more on these kind of tasks.

5 Basic Experiment on sysbench

In this section, we will try to use the original test bench brought by postgres to check the working ability of these two databases.
Considering that Gauss db was created out of postgresql 9.2 as we searched in the Internet. So we will also make test on postgresql 9.2 docker in this chapter and all the following chapters.
Here are some sample codes that will be used in the following test.

- initialize sysbench for postgres

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-user=postgres \
  --pgsql-password='2023Letmedo!' \
  --pgsql-db=postgres \
  --pgsql-host=localhost \
  --pgsql-port=25432 \
  --threads=4 \
  --tables=10 \
  --time=60 \
  prepare
```

- run sysbench on postgres

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-user=postgres \
  --pgsql-password='2023Letmedo!' \
  --pgsql-db=postgres \
  --pgsql-host=localhost \
  --pgsql-port=25432 \
  --threads=4 \
  --tables=10 \
  --time=60 \
  run
```

- clear sysbench on postgres

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-user=postgres \
  --pgsql-password='2023Letmedo!' \
  --pgsql-db=postgres \
  --pgsql-host=localhost \
  --pgsql-port=25432 \
  --threads=4 \
  --tables=10 \
  --time=60 \
  cleanup
```

- initialize sysbench for postgres:9.2

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=35432 \
  --pgsql-db=postgres \
  --pgsql-user=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  prepare
```

- run sysbench on postgres:9.2

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=35432 \
  --pgsql-db=postgres \
  --pgsql-user=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  run
```

- clear sysbench on postgres:9.2

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=35432 \
  --pgsql-db=postgres \
  --pgsql-user=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  clean up
```

- initialize sysbench for openGauss

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=15432 \
  --pgsql-user=gaussdb \
  --pgsql-password='@Sjq123456' \
  --pgsql-db=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  prepare
```

- run sysbench on OpenGauss

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=15432 \
  --pgsql-user=gaussdb \
  --pgsql-password='@Sjq123456' \
  --pgsql-db=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  run
```

- clear sysbench on openGauss

```
sysbench oltp_read_write \
  --db-driver=pgsql \
  --pgsql-host=localhost \
  --pgsql-port=15432 \
  --pgsql-user=gaussdb \
  --pgsql-password='@Sjq123456' \
  --pgsql-db=postgres \
  --threads=4 \
  --tables=10 \
  --time=60 \
  clean up
```

5.1 Test on sysbench on the default setting of database, mono-user

5.1.1 Design

despite the fact that there maybe difference in the default config setting, i will first do sysbench on all of them with default setting, i will set thread = 1, table = 100.
So I will run the code in the "mono_User_bench.sh"

5.1.2 Result

Here is the result of the postgres:latest

experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	488.18	9763.60	20.00
2	281.85	5637.07	20.00
3	269.74	5394.84	20.00
4	339.47	6789.32	20.00
5	275.28	5505.58	20.00
6	339.90	6798.01	20.00
7	277.11	5542.24	20.00
8	270.18	5403.57	20.00
9	274.10	5482.04	20.00
10	282.54	5650.71	20.00

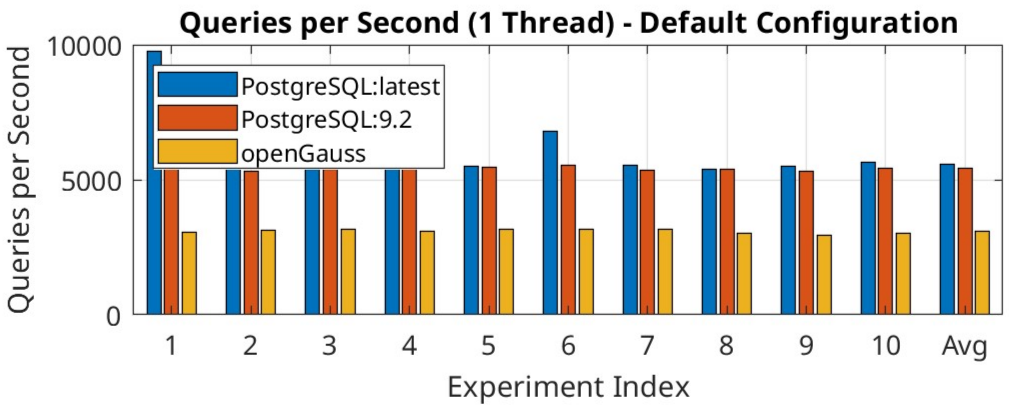
Here is the result of postgres:9.2

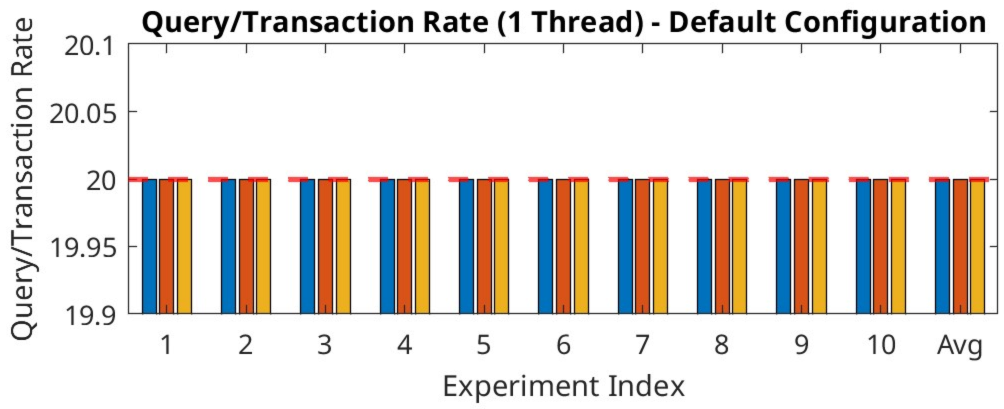
experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	271.46	5429.16	19.99
2	265.03	5300.67	20.00
3	268.53	5370.59	20.00
4	273.96	5479.24	20.00
5	272.87	5457.30	20.00
6	275.93	5518.68	20.00
7	267.99	5359.83	20.00
8	268.81	5376.23	20.00
9	265.88	5317.68	20.00
10	271.33	5426.64	20.00
Average	270.54	5417.87	20.00

Here is the solution of openGauss

experiment_index	transactions_per_sec	queries_per_sec	query_per_transaction_rate
1	152.73	3054.69	20.00
2	155.63	3112.67	20.00
3	158.10	3162.02	20.00
4	155.04	3100.75	20.00
5	158.58	3171.64	20.00
6	158.27	3165.41	20.00
7	158.48	3169.64	20.00
8	151.15	3022.97	20.00
9	147.26	2945.21	20.00
10	151.00	3020.01	20.00
Average	154.71	3097.73	20.00

We will have the graph to be





5.2 Test on sysbench on the default setting of database, 50-user

5.2.1 Design

despite the fact that there maybe difference in the default config setting, i will first do sysbench on all of them with default setting, i will set thread = 50, table = 100.
So I will run the code in the "50_User_bench.sh "

5.2.2 Result

Here is the result of the postgres:latest

experiment_index	duration	transactions_per_sec.	queries_per_sec.	query/transaction_rate
1	488.155	488.18	9763.60	20.0
2	488.384	281.85	5637.07	20.0
3	494.223	269.74	5394.84	20.0
4	485.179	339.47	6789.32	20.0
5	489.532	275.28	5505.58	20.0
6	497.237	339.90	6798.01	20.0
7	486.928	277.11	5542.24	20.0
8	488.424	270.18	5403.57	20.0
9	492.507	274.10	5482.04	20.0
10	483.013	282.54	5650.71	20.0
Average	293.84	5876.96	20.0	

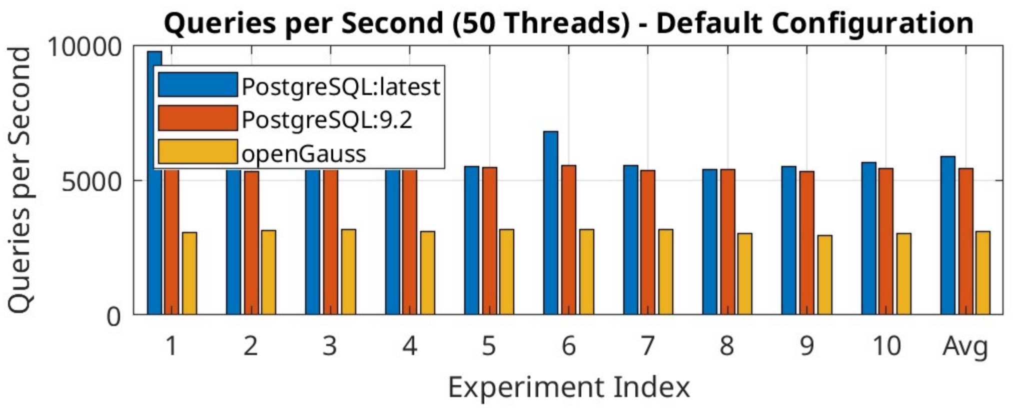
Here is the result of postgres:9.2

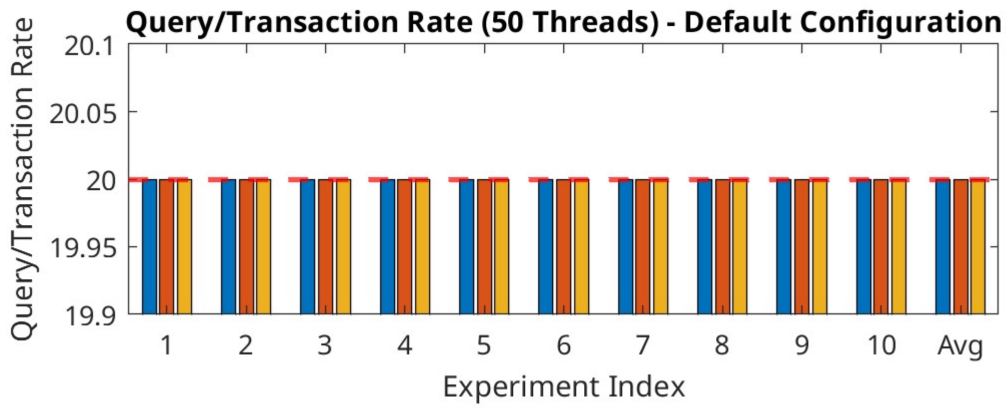
experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	271.46	5429.16	19.99
2	265.03	5300.67	20.00
3	268.53	5370.59	20.00
4	273.96	5479.24	20.00
5	272.87	5457.30	20.00
6	275.93	5518.68	20.00
7	267.99	5359.83	20.00
8	268.81	5376.23	20.00
9	265.88	5317.68	20.00
10	271.33	5426.64	20.00
Average	270.54	5417.87	20.00

Here is the solution of openGauss

experiment_index	transactions_per_sec	queries_per_sec	query_per_transaction_rate
1	152.73	3054.69	20.00
2	155.63	3112.67	20.00
3	158.10	3162.02	20.00
4	155.04	3100.75	20.00
5	158.58	3171.64	20.00
6	158.27	3165.41	20.00
7	158.48	3169.64	20.00
8	151.15	3022.97	20.00
9	147.26	2945.21	20.00
10	151.00	3020.01	20.00
Average	154.71	3097.73	20.00

We will have the graph to be





5.3 Test on sysbench on the unified loose setting of database, mono-user

5.3.1 Design

I will make the config parameter of these three database to be the same.
For the reason that the postgres:9.2 docker is based on too old version of Debian system, I find it is hard to have any text editor in the container. So i decide to change all the configurtion of three database to the one that of postgres:9.2 default.
And the major config will be listed as follows

- shared_buffers = 32MB
- temp_buffers = 8MB
- work_mem = 1MB
- maintenance_work_mem = 16MB
- max_stack_depth = 2MB

i will set thread = 1, table = 100.
So I will run the code in the "Mono_User_bench_unified.sh"

5.3.2 Result

Here is the result of the postgres:latest

experiment_index	transactions per second	queries per second	query/transaction rate
1	407.10	8141.98	20.00
2	257.34	5146.86	20.00
3	305.78	6115.60	20.00
4	252.66	5053.25	20.00
5	247.52	4950.39	20.00
6	313.26	6265.29	20.00
7	247.97	4959.34	20.00
8	245.89	4917.86	20.00
9	244.23	4884.60	20.00
10	245.86	4917.13	20.00
Average	270.97	5419.51	20.00

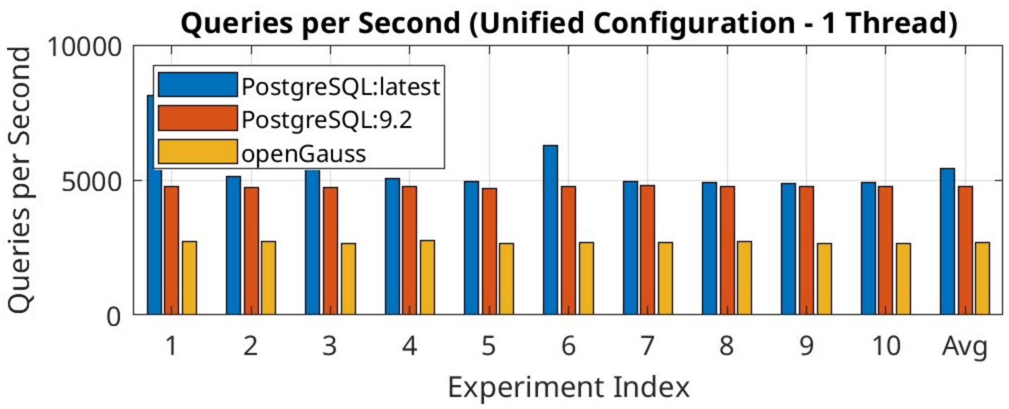
Here is the result of postgres:9.2

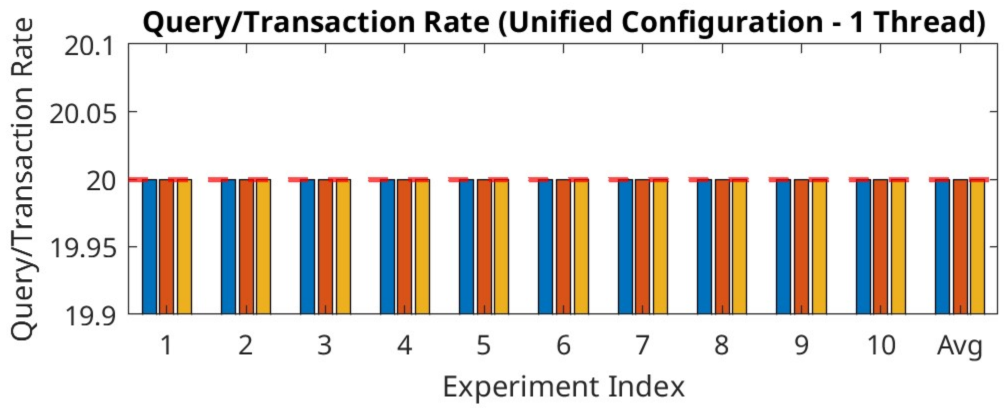
experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	238.56	4771.19	20.00
2	236.28	4725.54	20.00
3	237.01	4740.17	20.00
4	238.32	4766.38	20.00
5	235.13	4702.61	20.00
6	237.59	4751.90	20.00
7	239.33	4786.50	20.00
8	237.20	4744.10	20.00
9	237.40	4748.06	20.00
10	238.77	4775.45	20.00
Average	238.04	4757.02	20.00

Here is the solution of openGauss

experiment_index	transactions_per_sec (TPS)	queries_per_sec (QPS)	query_per_transaction_rate
1	135.74	2714.73	20.00
2	136.86	2737.11	20.00
3	133.27	2665.41	20.00
4	138.50	2769.97	20.00
5	131.87	2637.49	20.00
6	134.65	2693.00	20.00
7	135.10	2702.04	20.00
8	135.36	2707.22	20.00
9	132.17	2643.35	20.00
10	133.32	2666.49	20.00
Average	134.18	2697.58	20.00

We will have the graph to be





5.4 Test on sysbench on the unified loose setting of database, 50-user

5.4.1 Design

I will make the config parameter of these three database to be the same.
For the reason that the postgres:9.2 docker is based on too old version of Debian system, I find it is hard to have any text editor in the container. So i decide to change all the configurtion of three database to the one that of postgres:9.2 default.
And the major config will be listed as follows

- shared_buffers = 32MB
- temp_buffers = 8MB
- work_mem = 1MB
- maintenance_work_mem = 16MB
- max_stack_depth = 2MB

i will set thread = 50, table = 100.
So I will run the code in the "50_User_bench_unified.sh"

5.4.2 Result

Here is the result of the postgres:latest

experiment_index	transaction_per_sec	query_per_sec	query/transaction_rate
1	8334.09	168356.13	20.18
2	6892.82	139951.97	20.29
3	7482.49	151252.08	20.22
4	7930.13	160302.10	20.22
5	7580.74	153543.11	20.30
6	7917.08	160520.52	20.30
7	7971.90	161112.12	20.23
8	7209.65	145871.40	20.21
9	8275.89	167305.42	20.21
10	6620.08	134026.73	20.26
Average	7543.80	151569.08	20.25

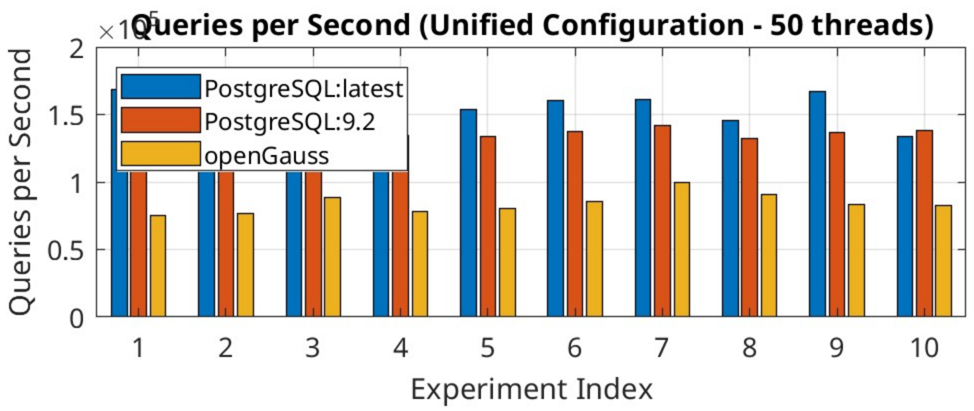
Here is the result of postgres:9.2

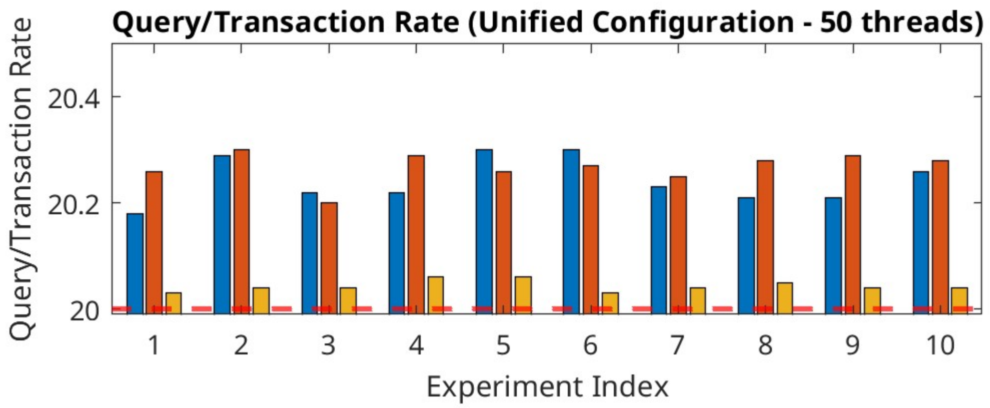
experiment_index	transactions_per_sec (TPS)	queries_per_sec (QPS)	query_per_transaction_rate
1	6902.43	139853.50	20.26
2	6679.02	135606.67	20.30
3	6588.23	133305.67	20.20
4	6653.95	134748.68	20.29
5	6611.61	133805.79	20.26
6	6770.99	137113.89	20.27
7	6998.97	141691.33	20.25
8	6534.30	132284.38	20.28
9	6729.08	136576.19	20.29
10	6814.45	138015.62	20.28
Average	6694.13	136801.43	20.27

Here is the solution of openGauss

experiment_index	transactions_per_sec (TPS)	queries_per_sec (QPS)	query_per_transaction_rate
1	3748.24	75134.63	20.03
2	3821.68	76605.80	20.04
3	4410.98	88435.09	20.04
4	3893.61	78037.94	20.06
5	3998.49	80142.88	20.06
6	4261.27	85429.07	20.03
7	4971.73	99674.77	20.04
8	4510.25	90428.29	20.05
9	4159.92	83360.49	20.04
10	4128.55	82756.12	20.04
Average	4215.58	84373.11	20.04

We will have the graph to be





5.5 Test on sysbench on the unified loose setting of database, 100-user

5.5.1 Design

I will make the config parameter of these three database to be the same.
For the reason that the postgres:9.2 docker is based on too old version of Debian system, I find it is hard to have any text editor in the container. So i decide to change all the configurtion of three database to the one that of postgres:9.2 default.
And the major config will be listed as follows

- shared_buffers = 32MB
- temp_buffers = 8MB
- work_mem = 1MB
- maintenance_work_mem = 16MB
- max_stack_depth = 2MB

i will set thread = 100, table = 100.
So I will run the code in the "100_User_bench_unified.sh"

5.5.2 Result

Here is the result of the postgres:latest

Experiment Index	Transactions per Second	Queries per Second	Queries/Transaction Rate
1	8762.76	177569.64	20.26
2	9597.50	194424.50	20.25
3	8852.80	180411.18	20.38
4	9952.11	202542.55	20.36
5	8900.59	180346.40	20.26
6	8011.15	164145.40	20.49
7	8305.65	169332.91	20.39
8	9328.66	190677.44	20.44
9	9976.18	202842.33	20.33
10	9782.03	198328.71	20.27
Average	9127.85	188522.86	20.33

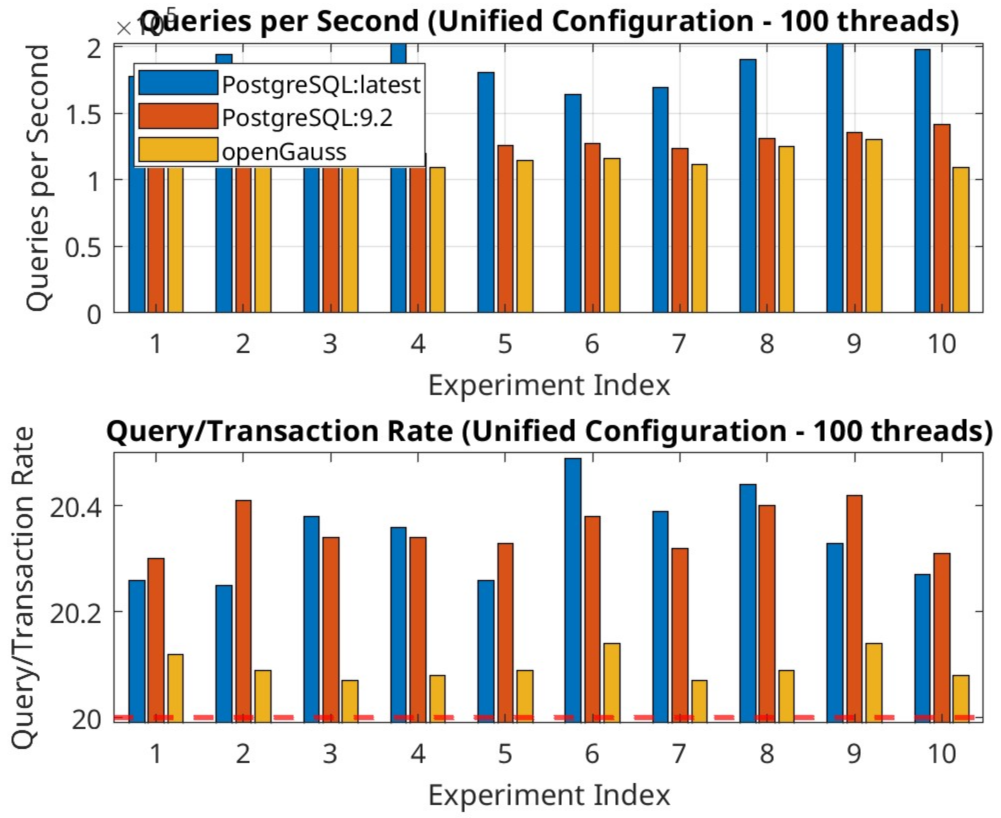
Here is the result of postgres:9.2

experiment_index	transactions/sec (TPS)	queries/sec (QPS)	query/transaction rate
1	6001.00	121831.89	20.30
2	6840.08	139586.76	20.41
3	6944.09	141139.07	20.34
4	5898.53	119939.02	20.34
5	6193.79	125890.93	20.33
6	6222.33	126965.36	20.38
7	6062.45	123310.19	20.32
8	6411.05	130821.25	20.40
9	6648.61	135742.48	20.42
10	6978.29	141672.76	20.31
average	6310.13	127679.94	20.35

Here is the solution of openGauss

experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	5976.27	120175.86	20.12
2	6093.69	122401.08	20.09
3	6166.73	123877.63	20.07
4	5428.95	109114.08	20.08
5	5693.63	114515.86	20.09
6	5754.91	115728.44	20.14
7	5550.68	111465.07	20.07
8	6204.32	124790.48	20.09
9	6473.80	130245.16	20.14
10	5422.24	108936.79	20.08
average	5880.53	117127.47	20.09

We will have the graph to be



6 Additional test on sysbench

In the result of the last chapter, we can see that the opengauss might have advantage on the ratio of qps/tps, so in this chapter, I will focus on this data, and i will use a bigger thread to test it, and at last i will explain why i think it is a notation that opengauss can handle transactions well when high threads occurs.
Because this is a big test, I use a bash file to do it , which is "night_test_all".
In this test, I keep the key config of the database postgres:latest and opengausss the same, because of the oldness of postgres:9.2 and which makes it hard to install any text editor in the docker, this time i have to give it up. I will just make a comparison between two databases.
I will use 3 tables and threads from 10 to 600

6.1 result

6.1.1 600 threads

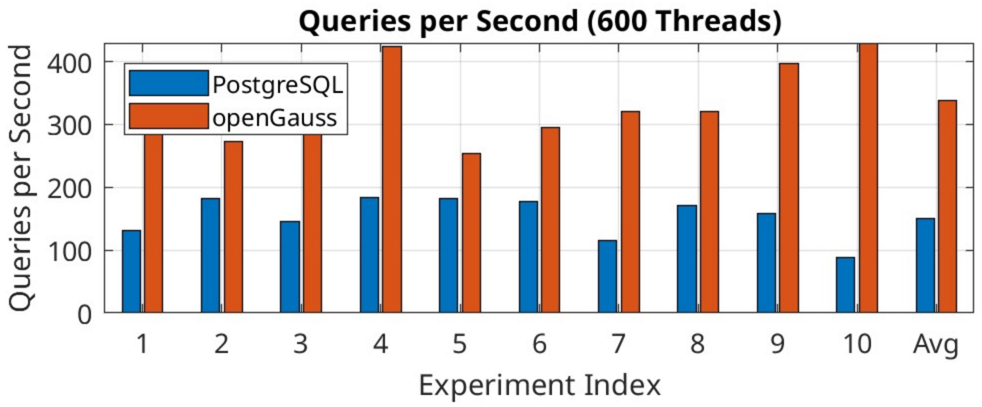
Here is the result of the postgres:latest

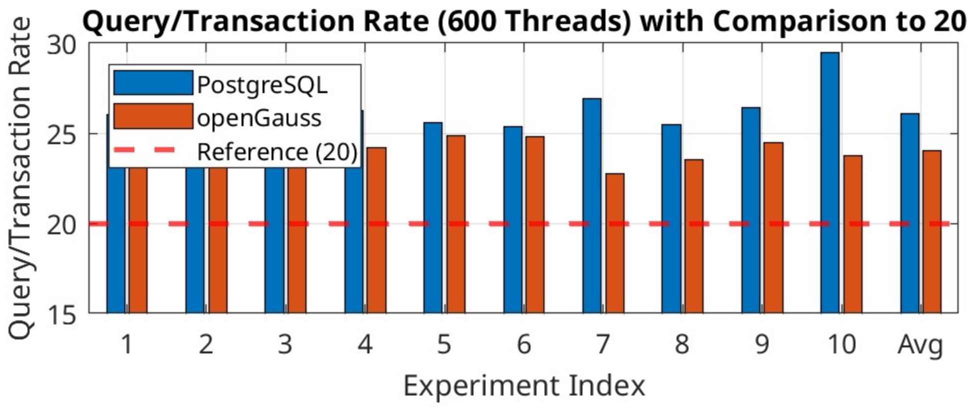
experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	5.03	131.08	26.05
2	7.02	182.57	26.04
3	5.57	145.41	26.12
4	6.99	183.87	26.26
5	7.10	181.44	25.57
6	6.97	176.60	25.34
7	4.27	115.04	26.92
8	6.74	171.56	25.46
9	5.98	157.98	26.42
10	3.02	89.07	29.5
Average	5.70	150.16	26.11

Here is the solution of openGauss

experiment_index	transactions_per_sec	queries_per_sec	query/transaction_rate
1	12.97	322.00	24.85
2	10.79	272.12	25.25
3	15.64	385.26	24.64
4	17.52	424.31	24.22
5	10.20	253.04	24.84
6	11.87	294.68	24.82
7	14.12	321.14	22.73
8	13.58	319.92	23.55
9	16.25	397.63	24.49
10	18.09	429.54	23.77
Average	14.02	337.54	24.01

We will have the graph to be





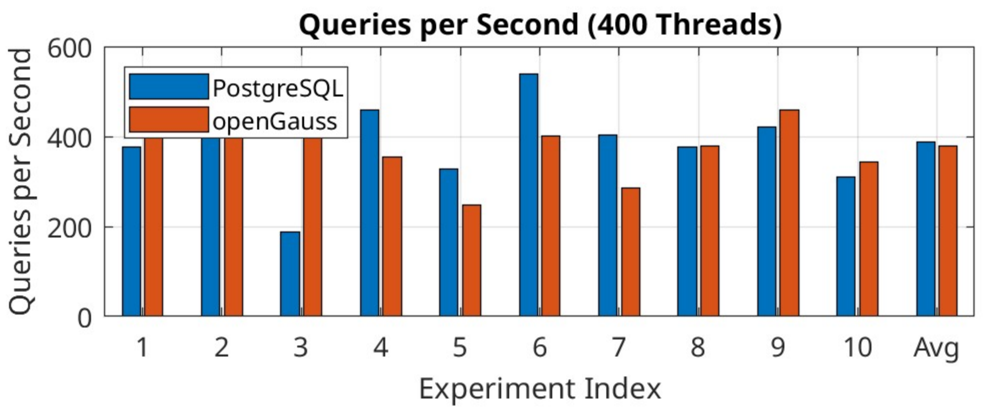
6.1.2 400 threads

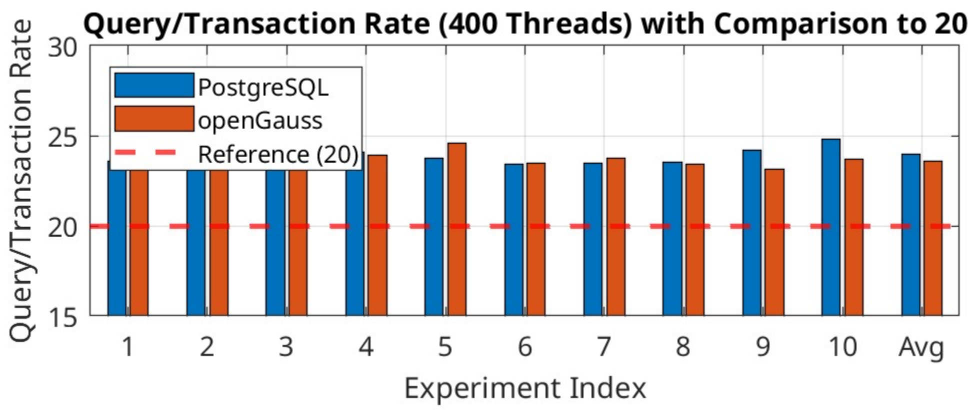
Here is the result of the postgres:latest

experiment_index	TPS (transactions per second)	QPS (queries per second)	QPS/TPS ratio
1	15.92	376.26	23.60
2	18.20	430.92	23.70
3	7.36	187.09	25.41
4	19.03	458.53	24.08
5	13.79	327.28	23.74
6	22.94	537.92	23.44
7	17.18	402.67	23.47
8	15.97	376.22	23.54
9	17.43	422.12	24.22
10	12.49	309.56	24.78
Average	16.04	387.60	24.00

Here is the solution of openGauss We will have the graph to be

experiment_index	TPS (transactions per second)	QPS (queries per second)	query/transaction rate
1	20.70	486.22	23.47
2	19.10	448.35	23.51
3	17.05	398.17	23.37
4	14.85	355.02	23.91
5	10.07	247.39	24.56
6	17.06	400.46	23.48
7	12.04	285.95	23.74
8	16.21	379.59	23.44
9	19.81	458.54	23.14
10	14.51	343.36	23.68
Average	16.53	378.05	23.56





6.1.3 200 threads

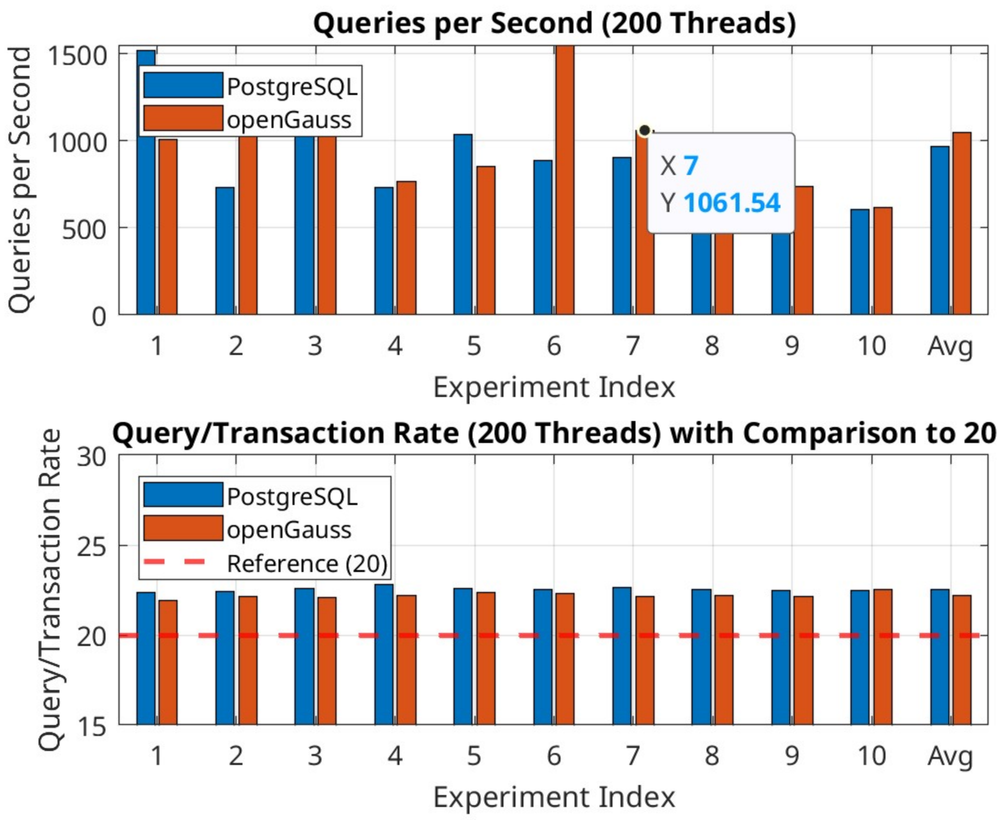
Here is the result of the postgres:latest

experiment_index	transaction_per_sec	query_per_sec	query/transaction_rate
1	68.10	1521.63	22.35
2	32.87	735.43	22.39
3	52.77	1192.90	22.59
4	31.99	730.65	22.83
5	45.94	1035.64	22.56
6	39.52	888.37	22.53
7	40.01	905.06	22.62
8	41.72	941.79	22.54
9	45.32	1017.16	22.47
10	27.01	605.96	22.47
Average	43.64	969.95	22.54

Here is the solution of openGauss

experiment_index	transaction_per_sec	query_per_sec	query/transaction_rate
1	45.94	1006.64	21.89
2	55.21	1223.40	22.14
3	61.39	1354.07	22.07
4	34.47	765.18	22.18
5	38.10	851.31	22.34
6	69.58	1551.63	22.33
7	48.04	1061.54	22.12
8	46.29	1030.15	22.22
9	33.33	737.99	22.14
10	27.30	614.92	22.51
Average	47.65	1051.48	22.21

We will have the graph to be



6.1.4 100 threads

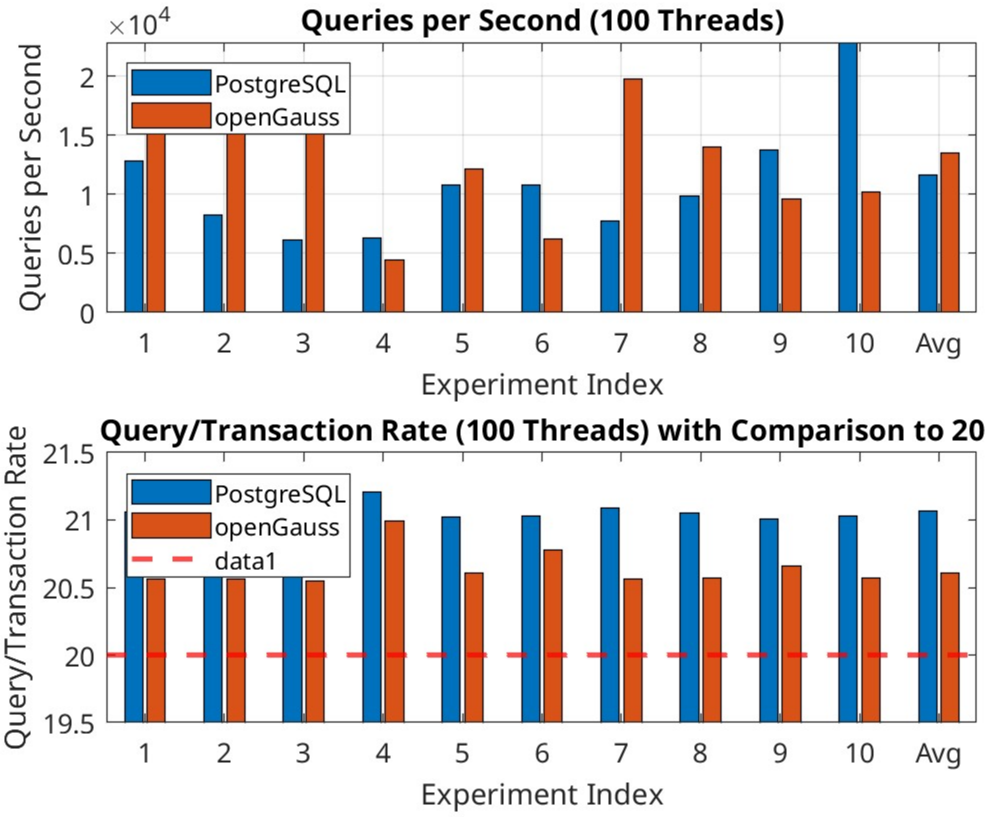
Here is the result of the postgres:latest

experiment_index	transactions per second	queries per second	query/transaction rate
1	611.21	12861.79	21.06
2	389.41	8219.52	21.12
3	291.37	6144.25	21.11
4	299.32	6343.79	21.21
5	513.06	10782.22	21.02
6	514.87	10818.73	21.03
7	368.60	7769.06	21.09
8	467.38	9836.74	21.05
9	654.21	13731.78	21.01
10	1089.47	22864.78	21.03
Average	551.29	11671.09	21.07

Here is the solution of openGauss

experiment_index	transaction_per_sec	query_per_sec	query/transaction_rate
1	776.23	15959.94	20.56
2	848.17	17444.91	20.56
3	903.98	18582.12	20.55
4	213.86	4487.06	20.99
5	590.36	12179.99	20.61
6	301.07	6247.41	20.78
7	964.52	19806.28	20.56
8	681.03	14005.50	20.57
9	463.38	9577.46	20.66
10	497.53	10237.08	20.57
Average	648.61	13467.19	20.61

We will have the graph to be



6.1.5 50 threads

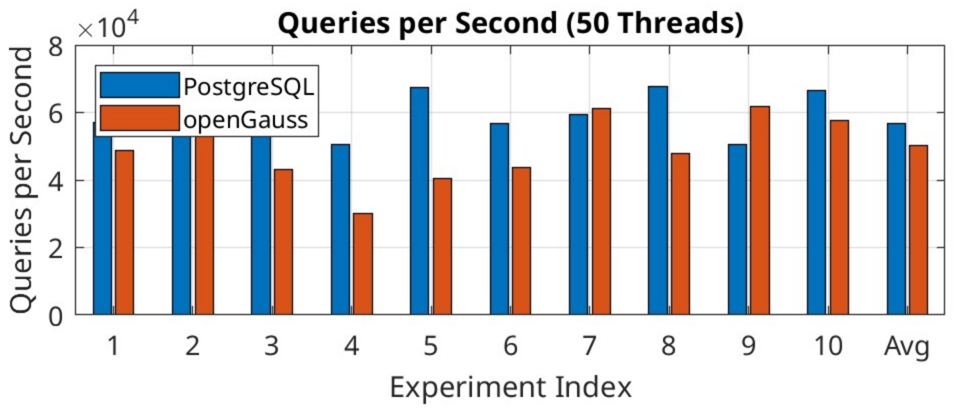
Here is the result of the postgres:latest

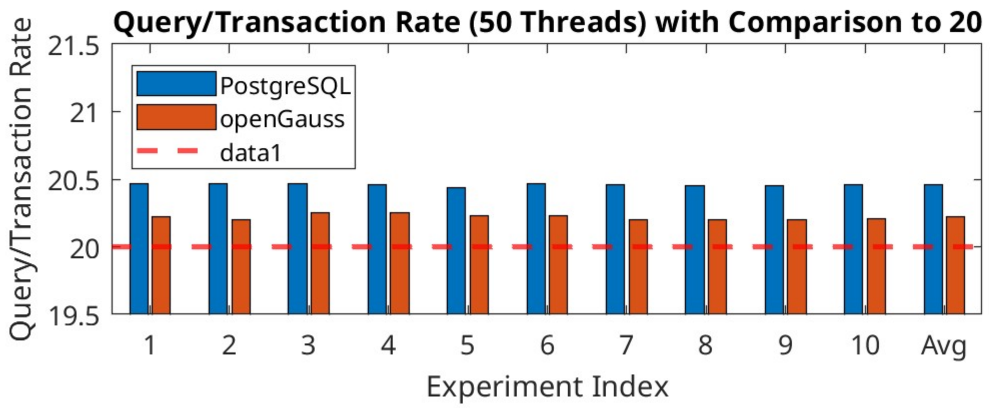
experiment_index	transactions per second	queries per second	query/transaction rate
1	2787.66	57033.65	20.47
2	2617.85	53558.52	20.47
3	2662.14	54433.58	20.47
4	2469.87	50514.69	20.46
5	3302.25	67499.51	20.44
6	2774.66	56738.59	20.47
7	2911.46	59531.10	20.46
8	3306.41	67598.13	20.45
9	2470.33	50544.87	20.45
10	3253.88	66565.52	20.46
average	2796.12	56673.51	20.46

Here is the solution of openGauss

experiment_index	transactions per second	queries per second	query/transaction rate
1	2417.09	48841.14	20.22
2	3035.49	61311.46	20.20
3	2132.63	43122.45	20.25
4	1480.71	29981.07	20.25
5	1997.04	40389.69	20.23
6	2166.01	43836.34	20.23
7	3024.10	61093.44	20.20
8	2367.61	47840.38	20.20
9	3062.03	61841.25	20.20
10	2847.75	57571.19	20.21
Average	2458.47	50133.61	20.22

We will have the graph to be





6.1.6 10 threads

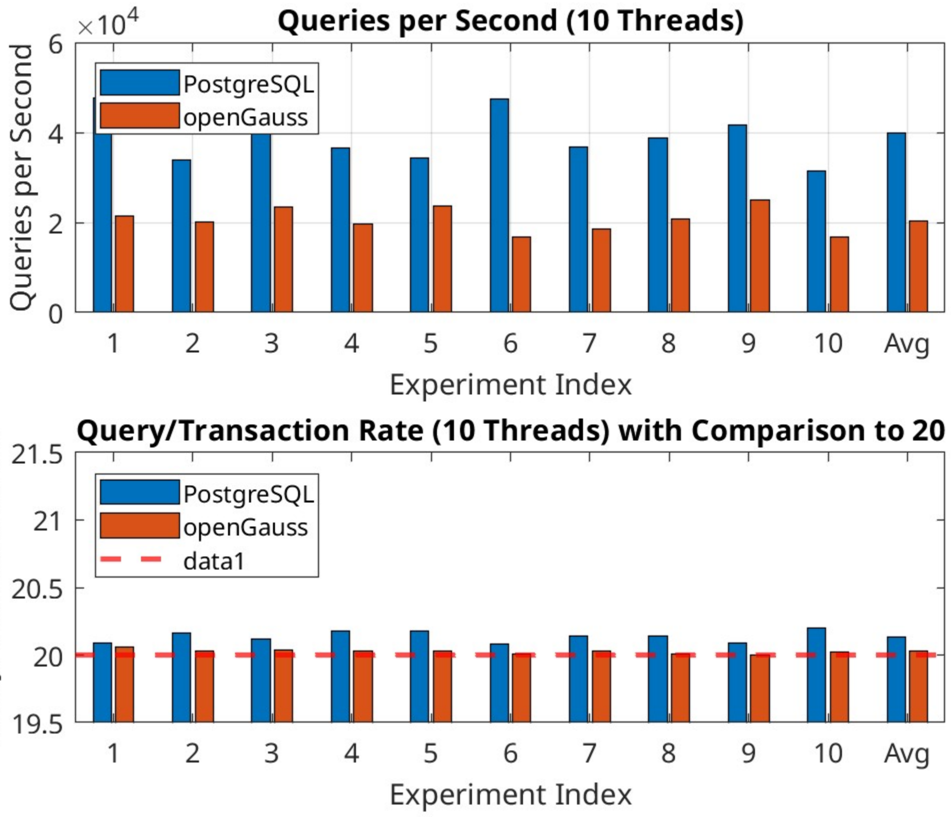
Here is the result of the postgres:latest

experiment_index	transactions_per_sec.	queries_per_sec.	query/transaction_rate
1	2366.36	47561.15	20.09
2	1683.64	33941.61	20.16
3	2646.97	53203.95	20.12
4	1810.39	36519.98	20.18
5	1703.99	34383.83	20.18
6	2356.95	47377.33	20.08
7	1822.62	36737.50	20.14
8	1926.13	38802.29	20.14
9	2072.60	41695.82	20.09
10	1555.56	31420.58	20.20
Average	1983.82	39778.53	20.13

Here is the solution of openGauss

experiment_index	transaction_per_sec	query_per_sec	query/transaction_rate
1	1074.31	21542.62	20.06
2	1006.43	20161.77	20.03
3	1165.24	23359.60	20.04
4	985.71	19765.38	20.03
5	1175.94	23569.14	20.03
6	835.46	16734.40	20.01
7	920.57	18454.36	20.03
8	1032.33	20683.54	20.01
9	1242.90	24905.66	20.00
10	842.60	16883.41	20.02
Average	1022.07	20439.13	20.03

We will have the graph to be



6.2 Conclusion

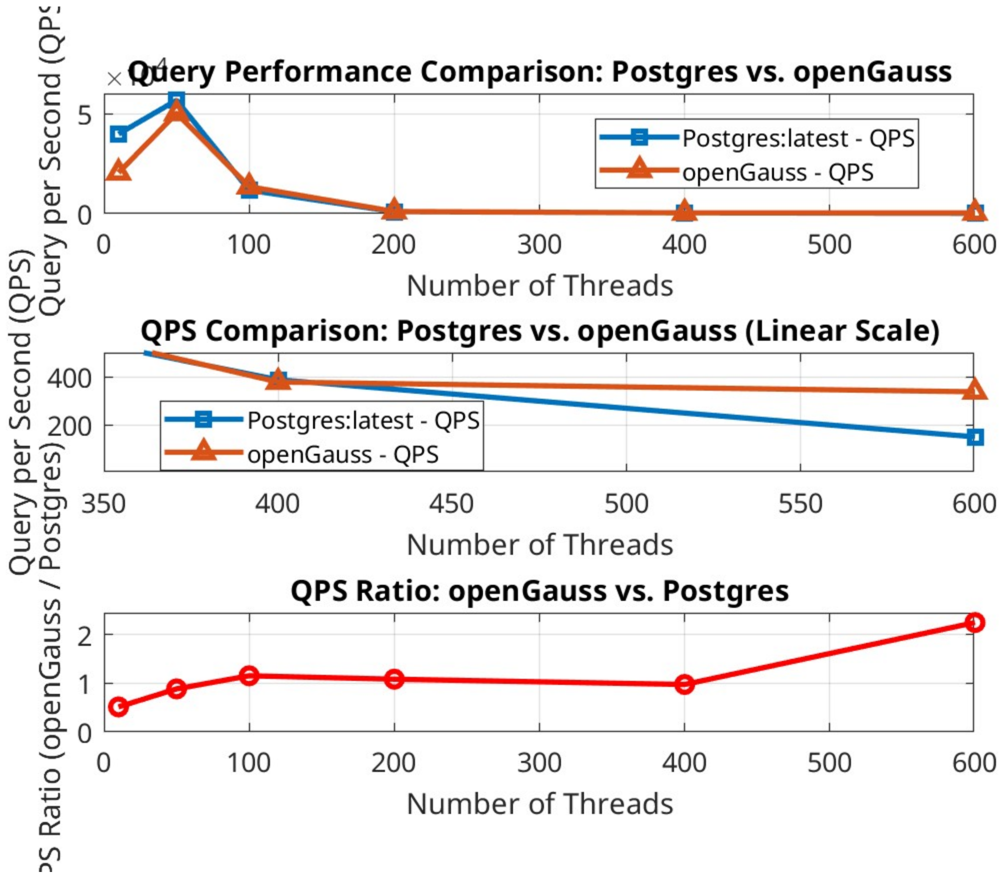
If we make a conclusion about the experiments above by the average result, we can find the following table. Here is the result of postgres:latest

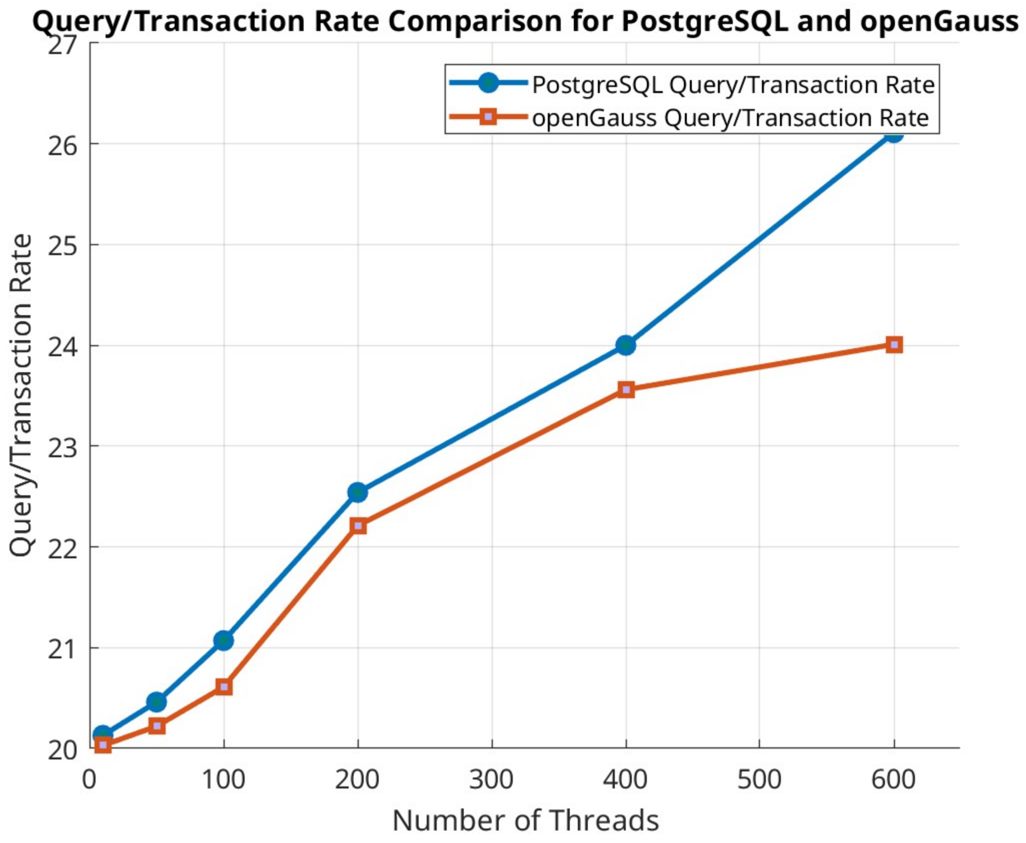
#thread	transaction_per_sec	query_per_sec	query/transaction_rate
10	1983.82	39778.53	20.13
50	2796.12	56673.51	20.46
100	551.29	11671.09	21.07
200	43.64	969.95	22.54
400	16.04	387.60	24.00
600	5.70	150.16	26.11

Here is the result of opneGauss

#thread	transaction_per_sec	query_per_sec	query/transaction_rate
10	1022.07	20439.13	20.03
50	2458.47	50133.61	20.22
100	648.61	13467.19	20.61
200	47.65	1051.48	22.21
400	16.53	378.05	23.56
600	14.02	337.54	24.01

And if you get a graph out of it, you will have





7 Pressure Test

7.1 Design

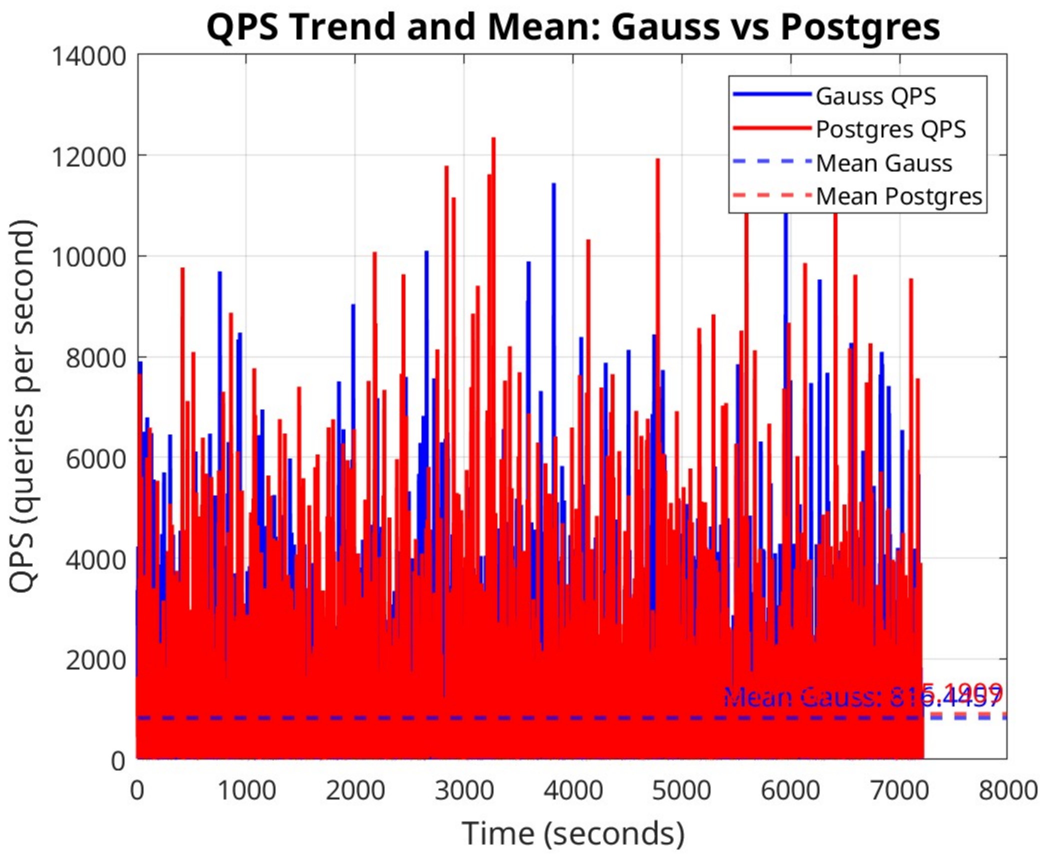
After we find that openGauss has some advantage when high number of threads occurs. I choose to offer a pressure test on the two databases used in the last chapter.
And I will have pressure test with basic parameters as follows:

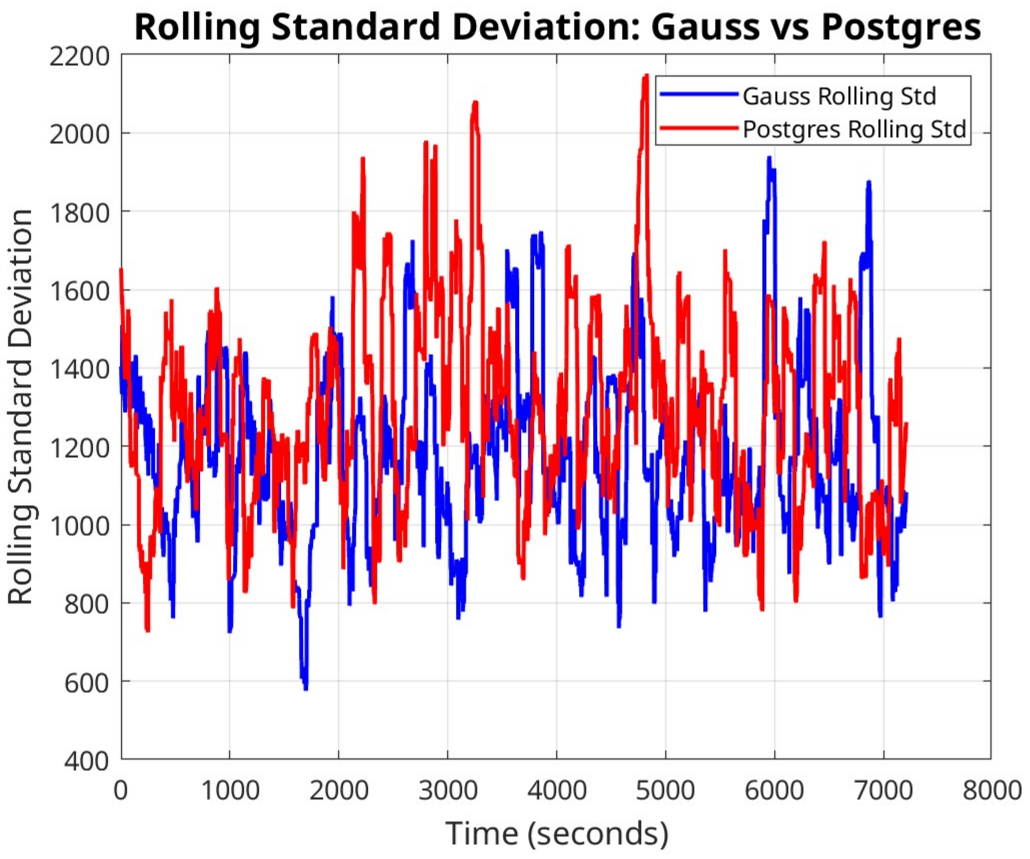
- threads = 500
- time = 7200

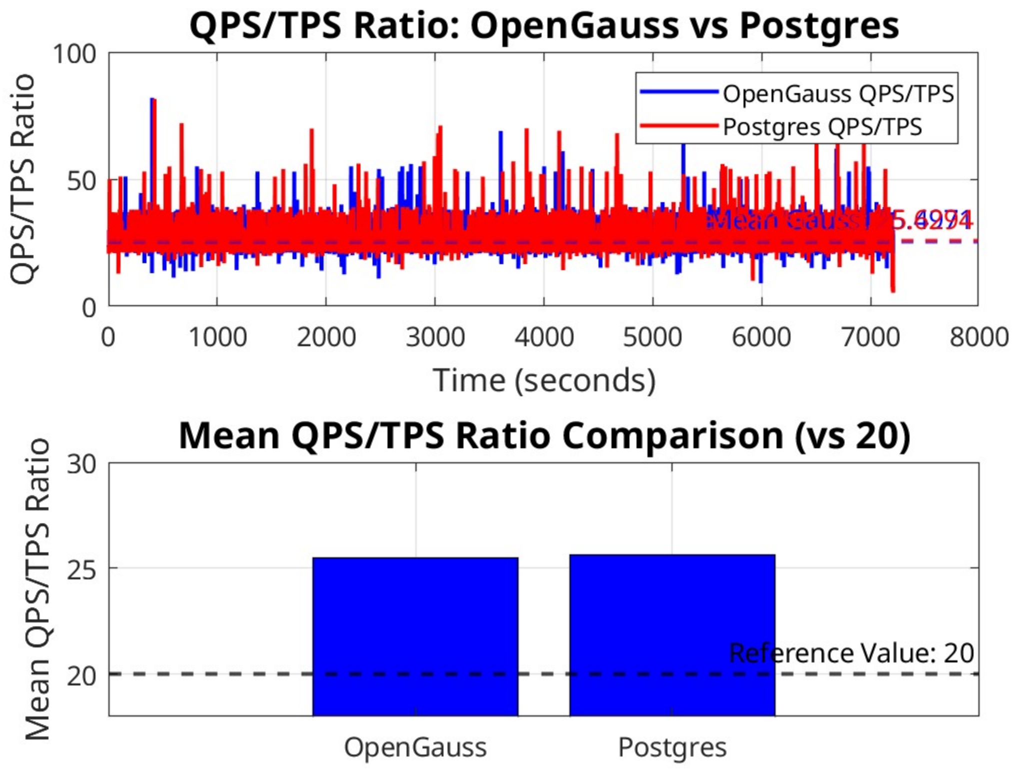
The code can be found also in "night_test_all.sh"

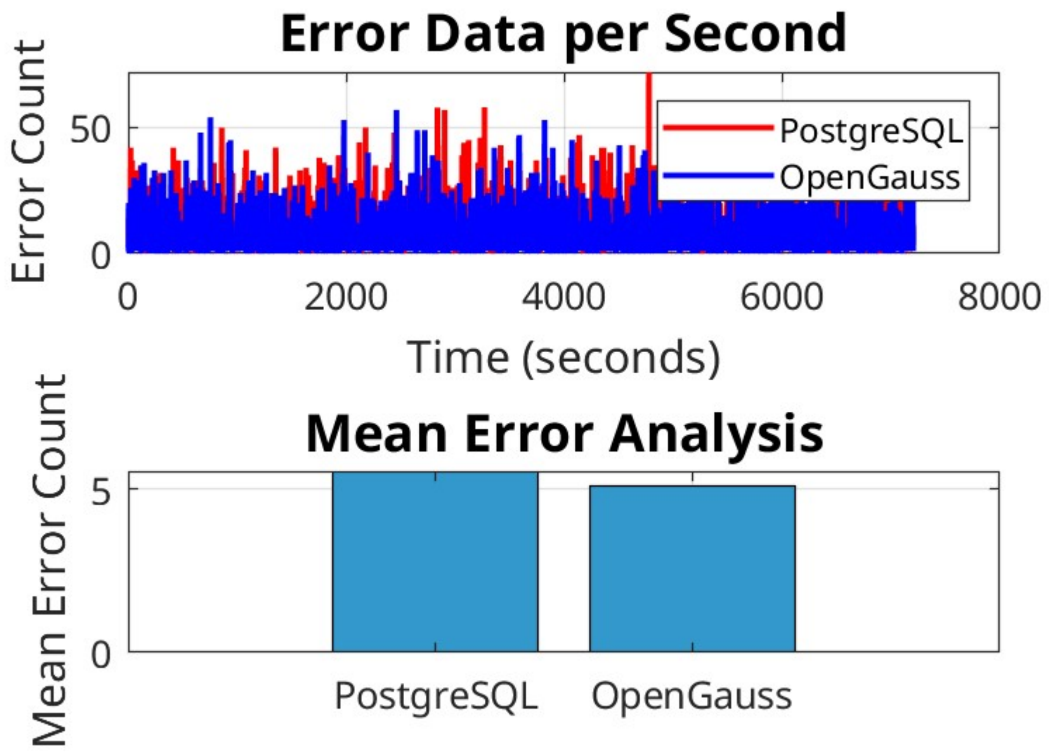
7.2 Result

I save the data in txt file, and i use matlab to plot the data directly.


QPS Trend and Mean: Gauss vs Postgres
QPS (queries per second)
Time (seconds)
Gauss QPS
Postgres QPS
Mean Gauss
Mean Postgres
Mean Gauss: 816.4497
5.1909
0
2000
4000
6000
8000
10000
12000
14000
1000
3000
5000
7000
8000


Rolling Standard Deviation: Gauss vs Postgres
Gauss Rolling Std
Postgres Rolling Std
Rolling Standard Deviation
Time (seconds)
2200
2000
1800
1600
1400
1200
1000
800
600
400
0
1000
2000
3000
4000
5000
6000
7000
8000


QPS/TPS Ratio: OpenGauss vs Postgres
QPS/TPS Ratio
Time (seconds)
OpenGauss QPS/TPS
Postgres QPS/TPS
Mean QPS/TPS Ratio Comparison (vs 20)
Mean QPS/TPS Ratio
Reference Value: 20
OpenGauss
Postgres



I have to explain that originally because of outstanding performance in 600 threads occasion, I really want to do pressure test on 600 threads, but opengause will break always. So that I change to test 500 threads instead. We know that from the previous experiments that opengauss has relatively pretty low qps when threads is low, and compared to postgres:latest, its qps is not drop as fast as postgres when threads number comes extremely high. And 500 threads is a number that these two databse system has similar qps.

With the similar value of qps, we can pay attention to other parameters. We can notice that QPS/TPS ratio is a parameter that openGauss has advantages, and it is not a surprise because we have already got this conclusion from the last chapter, but here we can vertify that this kind of advantage can hold for a long time under big pressure.

Also, we can see that under similar situation, we have openGauss system causes less error than postgres:latest, and that means that maybe opengauss have a more stable performance under big pressure for a long time, which is very important for the daily application for a database system.

8 Code

```
// import.sql
drop table if exists common_player_info cascade;
create table Common_Player_info(
person_id text,
first_name text,
last_name text,
display_first_last text,
display_last_comma_first text,
display_fi_last text,
player_slug text,
 birthdate text,
 school text,
 country text,
 last_affiliation text,
 height text,
 weight text,
 season_exp text,
 jersey text,
 position text,
 rosterstatus text,
 games_played_current_season_flag text,
 team_id text,
 team_name text,
 team_abbreviation text,
 team_code text,
 team_city text,
 playercode text,
 from_year text,
 to_year text,
 dleague_flag text,
 nba_flag text,
 games_played_flag text,
 draft_year text,
 draft_round text,
 draft_number text,
 greatest_75_flag text
 );
\copy Common_Player_info FROM '/tmp/trash/common_player_info.csv' DELIMITER ',' CSV HEADER;
 drop table if exists draft_combine_stats cascade;
 create table draft_combine_stats(
 season text,
 player_id text,
 first_name text,
 last_name text,
 player_name text,
 position text,
 height_wo_shoes text,
 height_wo_shoes_ft_in text,
 height_w_shoes text,
 height_w_shoes_ft_in text,
 weight text,
 wingspan text,
 wingspan_ft_in text,
 standing_reach text,
 standing_reach_ft_in text,
 body_fat_pct text,
 hand_length text,
 hand_width text,
 standing_vertical_leap text,
 max_vertical_leap text,
 lane_agility_time text,
 modified_lane_agility_time text,
 three_quarter_sprint text,
 bench_press text,
 spot_fifteen_corner_left text,
 spot_fifteen_break_left text,
 spot_fifteen_top_key text,
 spot_fifteen_break_right text,
 spot_fifteen_corner_right text,
 spot_college_corner_left text,
 spot_college_break_left text,
 spot_college_top_key text,
 spot_college_break_right text,
 spot_college_corner_right text,
 spot_nba_corner_left text,
 spot_nba_break_left text,
 spot_nba_top_key text,
 spot_nba_break_right text,
 spot_nba_corner_right text,
 off_drib_fifteen_break_left text,
 off_drib_fifteen_top_key text,
 off_drib_fifteen_break_right text,
 off_drib_college_break_left text,
 off_drib_college_top_key text,
 off_drib_college_break_right text,
 on_move_fifteen text,
 on_move_college text
 );
 \copy draft_combine_stats FROM '/tmp/trash/draft_combine_stats.csv' DELIMITER ',' CSV
     HEADER;
 drop table if exists draft_history cascade;
 create table draft_history(
 person_id text,
 player_name text,
 season text,
 round_number text,
 round_pick text,
 overall_pick text,
 draft_type text,
 team_id text,
 team_city text,
 team_name text,
 team_abbreviation text,
 organization text,
 organization_type text,
 player_profile_flag text
 );
 \copy draft_history FROM '/tmp/trash/draft_history.csv' DELIMITER ',' CSV HEADER;
 drop table if exists game cascade;
 create table game(
 season_id text,
 team_id_home text,
 team_abbreviation_home text,
 team_name_home text,
 game_id text,
 game_date text,
 matchup_home text,
 wl_home text,
 min text,
 fgm_home text,
 fga_home text,
 fg_pct_home text,
 fg3m_home text,
 fg3a_home text,
 fg3_pct_home text,
 ftm_home text,
 fta_home text,
 ft_pct_home text,
 oreb_home text,
 dreb_home text,
 reb_home text,
 ast_home text,
 stl_home text,
 blk_home text,
 tov_home text,
 pf_home text,
 pts_home text,
 plus_minus_home text,
 video_available_home text,
 team_id_away text,
 team_abbreviation_away text,
 team_name_away text,
 matchup_away text,
 wl_away text,
 fgm_away text,
 fga_away text,
 fg_pct_away text,
 fg3m_away text,
 fg3a_away text,
 fg3_pct_away text,
 ftm_away text,
 fta_away text,
 ft_pct_away text,
 oreb_away text,
 dreb_away text,
 reb_away text,
 ast_away text,
 stl_away text,
 blk_away text,
 tov_away text,
 pf_away text,
 pts_away text,
 plus_minus_away text,
 video_available_away text,
 season_type text
 );
 \copy game FROM '/tmp/trash/game.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists game_info cascade;
 create table game_info(
 game_id text,
 game_date text,
 attendance text,
 game_time text
 );
 \copy game_info FROM '/tmp/trash/game_info.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists game_summary cascade;
 create table game_summary(
 game_date_est text,
 game_sequence text,
 game_id text,
 game_status_id text,
 game_status_text text,
 gamecode text,
 home_team_id text,
 visitor_team_id text,
 season text,
 live_period text,
 live_pc_time text,
 natl_tv_broadcaster_abbreviation text,
 live_period_time_bcast text,
 wh_status text
 );
 \copy game_summary FROM '/tmp/trash/game_summary.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists inactive_players cascade;
 create table inactive_players(
 game_id text,
 player_id text,
 first_name text,
 last_name text,
 jersey_num text,
 team_id text,
 team_city text,
 team_name text,
 team_abbreviation text
 );
 \copy inactive_players FROM '/tmp/trash/inactive_players.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists line_score cascade;
 create table line_score(
 game_date_est text,
 game_sequence text,
 game_id text,
 team_id_home text,
 team_abbreviation_home text,
 team_city_name_home text,
 team_nickname_home text,
 team_wins_losses_home text,
 pts_qtr1_home text,
 pts_qtr2_home text,
 pts_qtr3_home text,
 pts_qtr4_home text,
 pts_ot1_home text,
 pts_ot2_home text,
 pts_ot3_home text,
 pts_ot4_home text,
 pts_ot5_home text,
 pts_ot6_home text,
 pts_ot7_home text,
 pts_ot8_home text,
 pts_ot9_home text,
 pts_ot10_home text,
 pts_home text,
 team_id_away text,
 team_abbreviation_away text,
 team_city_name_away text,
 team_nickname_away text,
 team_wins_losses_away text,
 pts_qtr1_away text,
 pts_qtr2_away text,
 pts_qtr3_away text,
 pts_qtr4_away text,
 pts_ot1_away text,
 pts_ot2_away text,
 pts_ot3_away text,
 pts_ot4_away text,
 pts_ot5_away text,
 pts_ot6_away text,
 pts_ot7_away text,
 pts_ot8_away text,
 pts_ot9_away text,
 pts_ot10_away text,
 pts_away text
 );
 \copy line_score FROM '/tmp/trash/line_score.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists officials cascade;
 create table officials(
 game_id text,
 official_id text,
 first_name text,
 last_name text,
 jersey_num text
 );
 \copy officials FROM '/tmp/trash/officials.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists other_stats cascade;
 create table other_stats(
 game_id text,
 league_id text,
 team_id_home text,
 team_abbreviation_home text,
 team_city_home text,
 pts_paint_home text,
 pts_2nd_chance_home text,
 pts_fb_home text,
 largest_lead_home text,
 lead_changes text,
 times_tied text,
 team_turnovers_home text,
 total_turnovers_home text,
 team_rebounds_home text,
 pts_off_to_home text,
 team_id_away text,
 team_abbreviation_away text,
 team_city_away text,
 pts_paint_away text,
 pts_2nd_chance_away text,
 pts_fb_away text,
 largest_lead_away text,
 team_turnovers_away text,
 total_turnovers_away text,
 team_rebounds_away text,
 pts_off_to_away text
 );
 \copy other_stats FROM '/tmp/trash/other_stats.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists play_by_play cascade;
 create table play_by_play(
 game_id text,
 eventnum text,
 eventmsgtype text,
 eventmsgactiontype text,
 period text,
 wctimestring text,
 pctimestring text,
 homedescription text,
 neutraldescription text,
 visitordescription text,
 score text,
 scoremargin text,
 person1type text,
 player1_id text,
 player1_name text,
 player1_team_id text,
 player1_team_city text,
 player1_team_nickname text,
 player1_team_abbreviation text,
 person2type text,
 player2_id text,
 player2_name text,
 player2_team_id text,
 player2_team_city text,
 player2_team_nickname text,
 player2_team_abbreviation text,
 person3type text,
 player3_id text,
 player3_name text,
 player3_team_id text,
 player3_team_city text,
 player3_team_nickname text,
 player3_team_abbreviation text,
 video_available_flag text
 );
 \COPY play_by_play from '/tmp/trash/play_by_play.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists player cascade;
 create table player(
 id text,
 full_name text,
 first_name text,
 last_name text,
 is_active text
 );
 \copy player FROM '/tmp/trash/player.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists team cascade;
 create table team(
 id text,
 full_name text,
 abbreviation text,
 nickname text,
 city text,
 state text,
 year_founded text
 );
 \copy team FROM '/tmp/trash/team.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists team_details cascade;
 create table team_details(
 team_id text,
 abbreviation text,
 nickname text,
 yearfounded text,
 city text,
 arena text,
 arenacapacity text,
 owner text,
 generalmanager text,
 headcoach text,
 dleagueaffiliation text,
 facebook text,
 instagram text,
 twitter text
 );
 \copy team_details FROM '/tmp/trash/team_details.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists team_history cascade;
 create table team_history(
 team_id text,
 city text,
 nickname text,
 year_founded text,
 year_active_till text
 );
 \copy team_history FROM '/tmp/trash/team_history.csv' DELIMITER ',' CSV HEADER;
 DROP table if exists team_info_common cascade;
 create table team_info_common(
 team_id text,
 season_year text,
 team_city text,
 team_name text,
 team_abbreviation text,
 team_conference text,
 team_division text,
 team_code text,
 team_slug text,
 w text,
 l text,
 pct text,
 conf_rank text,
 div_rank text,
 min_year text,
 max_year text,
 league_id text,
 season_id text,
 pts_rank text,
 pts_pg text,
 reb_rank text,
 reb_pg text,
 ast_rank text,
 ast_pg text,
 opp_pts_rank text,
 opp_pts_pg text
 );
 \copy team_info_common FROM '/tmp/trash/team_info_common.csv' DELIMITER ',' CSV HEADER;
```

```
// retrieval_selection.sql
SELECT * FROM play_by_play WHERE player1_name = 'Kevin Love';
```

```
// name_transformation.sql
UPDATE play_by_play SET player1_name =
REPLACE(player1_name, 'To', 'TTOO') WHERE player1_name LIKE '%To%';
```

```
// biconditional_selection.sql
SELECT * FROM play_by_play WHERE player1_name = 'Kevin Love'
AND game_id = '0021600215'
```

```
    // actual_retrieval_update.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/name_transformation.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import.sql
```

```
    // actual_biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/biconditional_selection.sql
```

```
    // import_plus_plus.sql
    drop table if exists common_player_info cascade;
create table Common_Player_info(
person_id text,
first_name text,
last_name text,
display_first_last text,
display_last_comma_first text,
display_fi_last text,
player_slug text,
birthdate text,
school text,
country text,
last_affiliation text,
height text,
weight text,
season_exp text,
jersey text,
position text,
rosterstatus text,
games_played_current_season_flag text,
team_id text,
team_name text,
team_abbreviation text,
team_code text,
team_city text,
playercode text,
from_year text,
to_year text,
dleague_flag text,
nba_flag text,
games_played_flag text,
draft_year text,
draft_round text,
draft_number text,
greatest_75_flag text
);
\copy Common_Player_info FROM '/tmp/trash/common_player_info.csv' DELIMITER ',' CSV HEADER;
drop table if exists draft_combine_stats cascade;
create table draft_combine_stats(
season text,
player_id text,
first_name text,
last_name text,
player_name text,
position text,
height_wo_shoes text,
height_wo_shoes_ft_in text,
height_w_shoes text,
height_w_shoes_ft_in text,
weight text,
wingspan text,
wingspan_ft_in text,
standing_reach text,
standing_reach_ft_in text,
body_fat_pct text,
hand_length text,
hand_width text,
standing_vertical_leap text,
max_vertical_leap text,
lane_agility_time text,
modified_lane_agility_time text,
three_quarter_sprint text,
bench_press text,
spot_fifteen_corner_left text,
spot_fifteen_break_left text,
spot_fifteen_top_key text,
spot_fifteen_break_right text,
spot_fifteen_corner_right text,
spot_college_corner_left text,
spot_college_break_left text,
spot_college_top_key text,
spot_college_break_right text,
spot_college_corner_right text,
spot_nba_corner_left text,
spot_nba_break_left text,
spot_nba_top_key text,
spot_nba_break_right text,
spot_nba_corner_right text,
off_drib_fifteen_break_left text,
off_drib_fifteen_top_key text,
off_drib_fifteen_break_right text,
off_drib_college_break_left text,
off_drib_college_top_key text,
off_drib_college_break_right text,
on_move_fifteen text,
on_move_college text
);
\copy draft_combine_stats FROM '/tmp/trash/draft_combine_stats.csv' DELIMITER ',' CSV HEADER;
drop table if exists draft_history cascade;
create table draft_history(
person_id text,
player_name text,
season text,
round_number text,
round_pick text,
overall_pick text,
draft_type text,
team_id text,
team_city text,
team_name text,
team_abbreviation text,
organization text,
organization_type text,
player_profile_flag text
);
\copy draft_history FROM '/tmp/trash/draft_history.csv' DELIMITER ',' CSV HEADER;
drop table if exists game cascade;
create table game(
season_id text,
team_id_home text,
team_abbreviation_home text,
team_name_home text,
game_id text,
game_date text,
matchup_home text,
wl_home text,
min text,
fgm_home text,
fga_home text,
fg_pct_home text,
fg3m_home text,
fg3a_home text,
fg3_pct_home text,
ftm_home text,
fta_home text,
ft_pct_home text,
oreb_home text,
dreb_home text,
reb_home text,
ast_home text,
stl_home text,
blk_home text,
tov_home text,
pf_home text,
pts_home text,
plus_minus_home text,
video_available_home text,
team_id_away text,
team_abbreviation_away text,
team_name_away text,
matchup_away text,
wl_away text,
fgm_away text,
fga_away text,
fg_pct_away text,
fg3m_away text,
fg3a_away text,
fg3_pct_away text,
ftm_away text,
fta_away text,
ft_pct_away text,
oreb_away text,
dreb_away text,
reb_away text,
ast_away text,
stl_away text,
blk_away text,
tov_away text,
pf_away text,
pts_away text,
plus_minus_away text,
video_available_away text,
season_type text
);
\copy game FROM '/tmp/trash/game.csv' DELIMITER ',' CSV HEADER;
DROP table if exists game_info cascade;
create table game_info(
game_id text,
game_date text,
attendance text,
game_time text
);
\copy game_info FROM '/tmp/trash/game_info.csv' DELIMITER ',' CSV HEADER;
DROP table if exists game_data_20000_rows cascade;
create table game_data_20000_rows(
game_id text,
game_date text,
attendance text,
game_time text
);
\copy game_data_20000_rows FROM '/tmp/trash/game_data_20000_rows.csv' DELIMITER ',' CSV HEADER;
DROP table if exists game_summary cascade;
create table game_summary(
game_date_est text,
game_sequence text,
game_id text,
game_status_id text,
game_status_text text,
gamecode text,
home_team_id text,
visitor_team_id text,
season text,
live_period text,
live_pc_time text,
natl_tv_broadcaster_abbreviation text,
live_period_time_bcast text,
wh_status text
);
```

```
\copy game_summary FROM '/tmp/trash/game_summary.csv' DELIMITER ',' CSV HEADER;
DROP table if exists inactive_players cascade;
create table inactive_players(
game_id text,
player_id text,
first_name text,
last_name text,
jersey_num text,
team_id text,
team_city text,
team_name text,
team_abbreviation text
);
\copy inactive_players FROM '/tmp/trash/inactive_players.csv' DELIMITER ',' CSV HEADER;
DROP table if exists line_score cascade;
create table line_score(
game_date_est text,
game_sequence text,
game_id text,
team_id_home text,
team_abbreviation_home text,
team_city_name_home text,
team_nickname_home text,
team_wins_losses_home text,
pts_qtr1_home text,
pts_qtr2_home text,
pts_qtr3_home text,
pts_qtr4_home text,
pts_ot1_home text,
pts_ot2_home text,
pts_ot3_home text,
pts_ot4_home text,
pts_ot5_home text,
pts_ot6_home text,
pts_ot7_home text,
pts_ot8_home text,
pts_ot9_home text,
pts_ot10_home text,
pts_home text,
team_id_away text,
team_abbreviation_away text,
team_city_name_away text,
team_nickname_away text,
team_wins_losses_away text,
pts_qtr1_away text,
pts_qtr2_away text,
pts_qtr3_away text,
pts_qtr4_away text,
pts_ot1_away text,
pts_ot2_away text,
pts_ot3_away text,
pts_ot4_away text,
pts_ot5_away text,
pts_ot6_away text,
pts_ot7_away text,
pts_ot8_away text,
pts_ot9_away text,
pts_ot10_away text,
pts_away text
```

```
);
\copy line_score FROM '/tmp/trash/line_score.csv' DELIMITER ',' CSV HEADER;
DROP table if exists officials cascade;
create table officials(
game_id text,
official_id text,
first_name text,
last_name text,
jersey_num text
);
\copy officials FROM '/tmp/trash/officials.csv' DELIMITER ',' CSV HEADER;
DROP table if exists other_stats cascade;
create table other_stats(
game_id text,
league_id text,
team_id_home text,
team_abbreviation_home text,
team_city_home text,
pts_paint_home text,
pts_2nd_chance_home text,
pts_fb_home text,
largest_lead_home text,
lead_changes text,
times_tied text,
team_turnovers_home text,
total_turnovers_home text,
team_rebounds_home text,
pts_off_to_home text,
team_id_away text,
team_abbreviation_away text,
team_city_away text,
pts_paint_away text,
pts_2nd_chance_away text,
pts_fb_away text,
largest_lead_away text,
team_turnovers_away text,
total_turnovers_away text,
team_rebounds_away text,
pts_off_to_away text
);
\copy other_stats FROM '/tmp/trash/other_stats.csv' DELIMITER ',' CSV HEADER;
DROP table if exists play_by_play cascade;
create table play_by_play(
game_id text,
eventnum text,
eventmsgtype text,
eventmsgactiontype text,
period text,
wctimestring text,
pctimestring text,
homedescription text,
neutraldescription text,
visitordescription text,
score text,
scoremargin text,
person1type text,
player1_id text,
player1_name text,
player1_team_id text,
player1_team_city text,
player1_team_nickname text,
player1_team_abbreviation text,
person2type text,
player2_id text,
player2_name text,
player2_team_id text,
player2_team_city text,
player2_team_nickname text,
player2_team_abbreviation text,
person3type text,
player3_id text,
player3_name text,
player3_team_id text,
player3_team_city text,
player3_team_nickname text,
player3_team_abbreviation text,
video_available_flag text
);
\COPY play_by_play from '/tmp/trash/play_by_play.csv' DELIMITER ',' CSV HEADER;
DROP table if exists player cascade;
create table player(
id text,
full_name text,
first_name text,
last_name text,
is_active text
);
\copy player FROM '/tmp/trash/player.csv' DELIMITER ',' CSV HEADER;
DROP table if exists team cascade;
create table team(
id text,
full_name text,
abbreviation text,
nickname text,
city text,
state text,
year_founded text
);
\copy team FROM '/tmp/trash/team.csv' DELIMITER ',' CSV HEADER;
DROP table if exists team_details cascade;
create table team_details(
team_id text,
abbreviation text,
nickname text,
yearfounded text,
city text,
arena text,
arenacapacity text,
owner text,
generalmanager text,
headcoach text,
dleagueaffiliation text,
facebook text,
instagram text,
twitter text
);
\copy team_details FROM '/tmp/trash/team_details.csv' DELIMITER ',' CSV HEADER;
DROP table if exists team_history cascade;
```

```
create table team_history(
team_id text,
city text,
nickname text,
year_founded text,
year_active_till text
);
\copy team_history FROM '/tmp/trash/team_history.csv' DELIMITER ',' CSV HEADER;
DROP table if exists team_info_common cascade;
create table team_info_common(
team_id text,
season_year text,
team_city text,
team_name text,
team_abbreviation text,
team_conference text,
team_division text,
team_code text,
team_slug text,
w text,
l text,
pct text,
conf_rank text,
div_rank text,
min_year text,
max_year text,
league_id text,
season_id text,
pts_rank text,
pts_pg text,
reb_rank text,
reb_pg text,
ast_rank text,
ast_pg text,
opp_pts_rank text,
opp_pts_pg text
);
\copy team_info_common FROM '/tmp/trash/team_info_common.csv' DELIMITER ',' CSV HEADER;
```

```
    // actual_large_scale_insert.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/insert_table_by_select_table_big.sql
    \i /home/sjq/Documents/Assignment/CS213/ProjectThree/import_plus_plus.sql
```

```
// Mono_User_bench.sh
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/mono_user_default_postgres.log"
LOGFILE2="/home/sjq/Documents/Assignment/CS213/ProjectThree/mono_user_default_postgres9_2.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/mono_user_default_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE2" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
```

```
\\50_User_bench.sh
```

```
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_default_postgres.log"
LOGFILE2="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_default_postgres9_2.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_default_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE2" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
```

```
//Mono_User_bench_unified.sh
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/unified_mono_user_default_postgres.log"
LOGFILE2="/home/sjq/Documents/Assignment/CS213/ProjectThree/unified_mono_user_default_postgres9_2.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/unified_mono_user_default_openGauss.log"
for number in {1..10}
do
```

```
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE2" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=1 \
        --tables=10 \
        --time=60 \
        cleanup
done
```

```
//50_User_bench_unified.sh
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_unified_postgres.log"
LOGFILE2="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_unified_postgres9_2.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_user_unified_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE2" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=10 \
        --time=60 \
        cleanup
done
```

```
//100_User_bench_unified.sh
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/100_user_unified_postgres.log"
LOGFILE2="/home/sjq/Documents/Assignment/CS213/ProjectThree/100_user_unified_postgres9_2.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/100_user_unified_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE2" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=35432 \
        --pgsql-db=postgres \
        --pgsql-user=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=10 \
        --time=60 \
        cleanup
done
```

```
//night_test_all.sh
#!/bin/bash

./night_test_2.sh
./night_test_3.sh
./night_test.sh
```

```
//night_test.sh
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/600_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/600_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=600 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=600 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=600 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=600 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=600 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=600 \
        --tables=3 \
        --time=20 \
        cleanup
done
```

```
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/400_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/400_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=400 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=400 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=400 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=400 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=400 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=400 \
        --tables=3 \
        --time=20 \
        cleanup
done
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/200_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/200_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=200 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=200 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=200 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=200 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=200 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=200 \
        --tables=3 \
        --time=20 \
        cleanup
done
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/100_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/100_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=100 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=100 \
        --tables=3 \
        --time=20 \
        cleanup
done
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/50_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=50 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=50 \
        --tables=3 \
        --time=20 \
        cleanup
done
# Log file
LOGFILE1="/home/sjq/Documents/Assignment/CS213/ProjectThree/10_3_20_postgres.log"
LOGFILE3="/home/sjq/Documents/Assignment/CS213/ProjectThree/10_3_20_openGauss.log"
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=10 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=10 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE1" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-user=postgres \
        --pgsql-password='2023Letmedo!' \
        --pgsql-db=postgres \
        --pgsql-host=localhost \
        --pgsql-port=25432 \
        --threads=10 \
        --tables=3 \
        --time=20 \
        cleanup
done
for number in {1..10}
do
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=10 \
        --tables=3 \
        --time=20 \
        prepare
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=10 \
        --tables=3 \
        --time=20 \
        run >> "$LOGFILE3" 2>&1
    sysbench oltp_read_write \
        --db-driver=pgsql \
        --pgsql-host=localhost \
        --pgsql-port=15432 \
        --pgsql-user=gaussdb \
        --pgsql-password='@Sjq123456' \
        --pgsql-db=postgres \
        --threads=10 \
        --tables=3 \
        --time=20 \
        cleanup
done
```

```
//night_test_2.sh
#!/bin/bash

# Files to store the extracted data
tps_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/tps_data_postgres.txt"
qps_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/qps_data_postgres.txt"
err_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/err_data_postgres.txt"

# Clear previous data in the files
> "$tps_file"
> "$qps_file"
> "$err_file"

# Prepare the database
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=25432 \
    --pgsql-user=postgres \
    --pgsql-password='2023Letmedo!' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    prepare

# Run the test and extract the data
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=25432 \
    --pgsql-user=postgres \
    --pgsql-password='2023Letmedo!' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    run | while IFS= read -r line; do
    # Extract ‘tps‘, ‘qps‘, and ‘err/s‘ using awk
    tps=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="tps:") print $(i+1)}')
    qps=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="qps:") print $(i+1)}')
    err=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="err/s:") print $(i+1)}')

    # Append the extracted values to their respective files if they are not empty
    if [[ -n "$tps" ]]; then echo "$tps" >> "$tps_file"; fi
    if [[ -n "$qps" ]]; then echo "$qps" >> "$qps_file"; fi
    if [[ -n "$err" ]]; then echo "$err" >> "$err_file"; fi
done

# Cleanup the database
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=25432 \
    --pgsql-user=postgres \
    --pgsql-password='2023Letmedo!' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    cleanup
```

```
//night_test_3.sh
#!/bin/bash

# Files to store the extracted data
tps_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/tps_data_gauss.txt"
qps_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/qps_data_gauss.txt"
err_file="/home/sjq/Documents/Assignment/CS213/ProjectThree/err_data_gauss.txt"

# Clear previous data in the files
> "$tps_file"
> "$qps_file"
> "$err_file"

# Prepare the database
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=15432 \
    --pgsql-user=gaussdb \
    --pgsql-password='@Sjq123456' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    prepare

# Run the test and extract the data
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=15432 \
    --pgsql-user=gaussdb \
    --pgsql-password='@Sjq123456' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    run | while IFS= read -r line; do
    # Extract `tps`, `qps`, and `err/s` using awk
    tps=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="tps:") print $(i+1)}')
    qps=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="qps:") print $(i+1)}')
    err=$(echo "$line" | awk -F " " '{for(i=1;i<=NF;i++) if($i=="err/s:") print $(i+1)}')

    # Append the extracted values to their respective files if they are not empty
    if [[ -n "$tps" ]]; then echo "$tps" >> "$tps_file"; fi
    if [[ -n "$qps" ]]; then echo "$qps" >> "$qps_file"; fi
    if [[ -n "$err" ]]; then echo "$err" >> "$err_file"; fi
done

# Cleanup the database
sysbench oltp_read_write \
    --db-driver=pgsql \
    --pgsql-host=localhost \
    --pgsql-port=15432 \
    --pgsql-user=gaussdb \
    --pgsql-password='@Sjq123456' \
    --pgsql-db=postgres \
    --threads=500 \
    --tables=10 \
    --time=7200 \
    --report-interval=1\
    cleanup
```